NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

THE SIMULATION OF REMOTELY MEASURED PATHS OF UNDERWATER VEHICLES FOR THE PURPOSE OF MONITORING THE CALIBRATION OF TEST RANGES

by

Gene Gygax

September 1985

Thesis Advisor: R. R. Read

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER	2. GOVT ACCESSION NO. / 3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle): The Simulation of Remotely Measured Paths of Underwater Vehicles for the Purpose of Monitoring the Calibration of Test Ranges	5. TYPE OF REPORT & PERIOD COVERED: Master's Thesis, September, 1985
	6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s): Gene Gygax	8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Naval Postgraduate School, Monterey, California 93943-5100	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS: Naval Postgraduate School, Monterey, California 93943-5100	12. REPORT DATE: September, 1985
	13. NUMBER OF PAGES: 107
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)	15. SECURITY CLASS. (of this report)
	15a. DECLASSIFICATION/ DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution is unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Simulation, Short Baseline Sonar Array, 3-D Path Simulation

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This thesis analyses data of and builds a simulation model for the track of an underwater vehicle as perceived by a test range of three dimensional short baseline sonar arrays. In this way many random replications of track become available quickly and inexpensively. These simulations support a larger project whose object is to monitor the performance of the test range and provide clues for troubleshooting problems. In particular, joint values

20. of sensor array estimated displacement and reorientation corrections are generated and their error distribution is quantified.

The Simulation of Remotely Measured Paths of Underwater Vehicles for the Purpose of Monitoring the Calibration of Test Ranges

by

Gene Gygax
Lieutenant Commander, United States Navy
B.S., United States Naval Academy, 1976

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL
September 1985

ABSTRACT

This thesis analyses data of and builds a simulation model for the track of an underwater vehicle as perceived by a test range of three dimensional short baseline sonar arrays. In this way many random replications of track become available quickly and inexpensively. These simulations support a larger project whose object is to monitor the performance of the test range and provide clues for troubleshooting problems. In particular, joint values of sensor array estimated displacement and reorientation corrections are generated and their error distribution is quantified.

TABLE OF CONTENTS

LIST OF FIGURES

I. INTRODUCTION

The Naval Undersea Weapons Engineering Station (NUWES) operates nine test ranges that serve a variety of purposes for testing and validating the Navy's current and future weapons systems. This thesis is in support of a larger project whose goal is to monitor the performance of the short baseline ranges (ie. those ranges in which each array produces a three dimensional track of moving vehicles). More specifically, it deals with the development of a computer simulation which generates realistic random replications of an underwater vehicle's track (ie. vehicle position perceived by a sensor array on the range at a number of equally spaced time points). Such simulations serve to provide information about the inherent variability in the output of the tracking range, especially for assessing the calibration error. Its use can provide an improved understanding of the processes involved and a reduction in the frequency of range shutdown for the purpose of resurveying the remote sensor locations.

When the simulated replicate tracks are passed to the program KEYMAIN (a project FORTRAN program that estimates displacement and orientation corrections to the range remote sensor arrays), the output is used to generate bivariate scatter plots of these corrections. Although extensive work

of this type was not possible in the current thesis, the limited results indicate a surprising amount of variability and suggest that the nature and extent of variability can change noticeably with relatively minor changes in the localized conditions. Considerable testing using this simulation tool is clearly indicated.

The research reported here involves three general activities:

(a) Analysis of real underwater track data to learn their important behavioral characteristics.

(b) Simulation model formulation and construction.

(c) Development and programming of algorithms to merge with the existing project programs to produce any number of random replications with correction estimates for each simulated track.

The organization of the thesis is as follows: Section II contains explicit background material and provides a framework for the research and shows how it relates to the larger project. Section III explains the data analysis procedures and results. The simulation model is developed in Section IV and the interfacing with the overall project programs is described in Section V. Results and conclusions are detailed in Section VI, with areas for future work listed in Section VII.

A number of appendices are included to provide the detailed support too extensive to be included in the main body of the paper. They are referenced in the appropriate sections. Additionally, to speed the completion and

availability of the KEYMAIN program, the author wrote two subroutines (CONECT and REDUCE) to be included in that package. Appendices E through G contain the development of these subroutines and the FORTRAN code listings.

II. BACKGROUND

Consider a three dimensional underwater tracking array composed of four hydrophones arranged to define a local Cartesian coordinate system (see Figure 1). Such an array is capable of tracking a target downrange, crossrange and in depth in its local coordinate system; it is termed a short baseline array because of the short distance of the X, Y, and Z hydrophones from the "corner" hydrophone (30 feet) that is used to gather the three dimensional tracking information.

The arrays take "fixes" of target position. A fix is defined by a bearing (azimuth and elevation) and distance from the array to the target. Tracked targets on the range are fitted with an acoustical device, called a "pinger", that emits pulses of sound, or "pings", at a specific frequency at precisely timed, regular intervals. The elapsed time from the generation of the ping at the target to the reception of the ping at the array establishes a distance from the target to the array. Because of the distance that separates the individual hydrophones of the array, the ping arrives at each hydrophone at a slightly different time. This time difference can be resolved into a bearing from the array to target. (Actually, these fixes are "apparent" fixes. They are used to initialize a sound

SCHEMATIC DIAGRAM OF 3-D SENSOR ARRAY

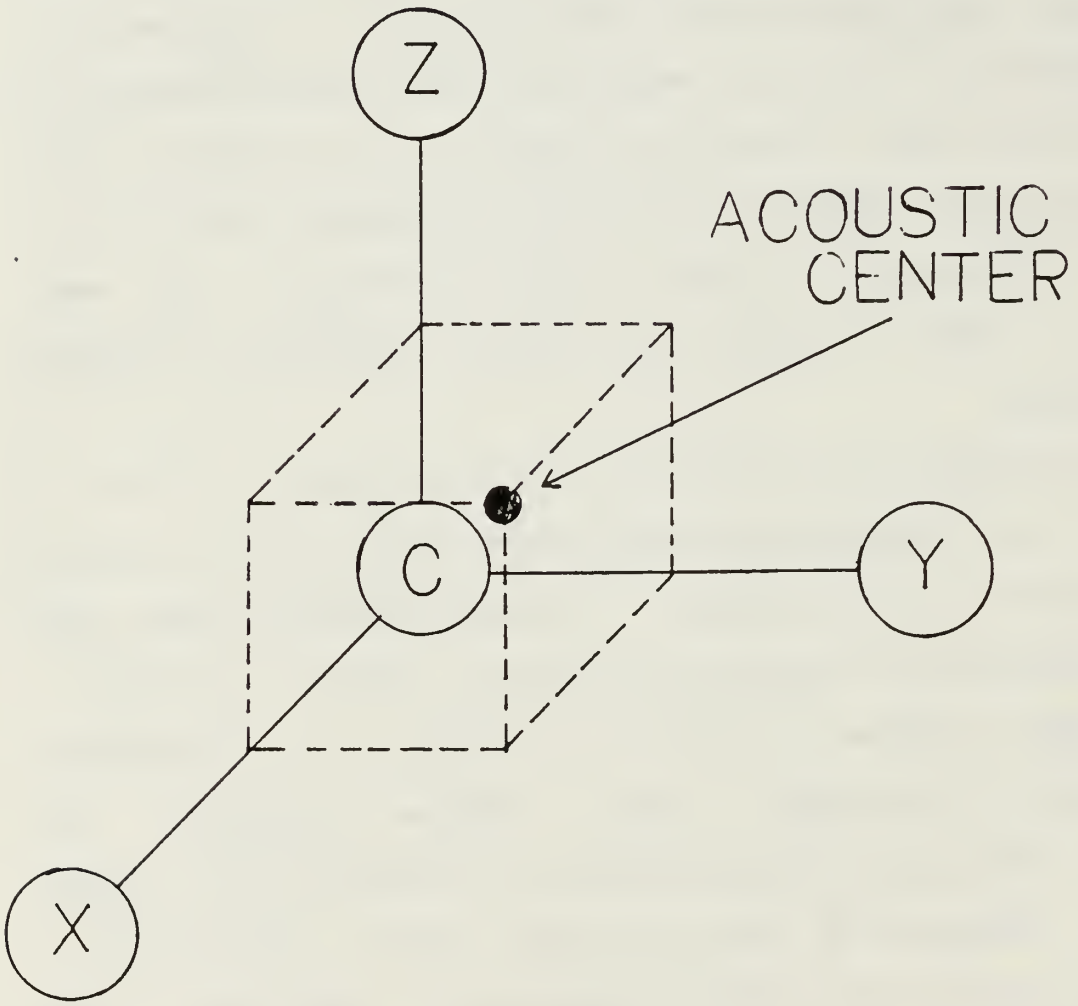


Figure 1 : 3-D Sensor Array

ray tracing algorithm that leads to the "real" fix.) A series of such fixes establishes a target track in that particular array's own local Cartesian coordinate system. If the precise position of the array is known, its local track information can be translated to one common range coordinate system.

Position of an individual array in the range Cartesian coordinate system is described not only in terms of its downrange, crossrange and depth (X,Y and Z) coordinates (termed location), but also with respect to X-tilt, Y-tilt and Z-rotation (commonly called roll, pitch and yaw) angles from the range coordinate system axes (termed orientation). Both sets of measures are needed to translate accurately from local to range coordinates.

Typically, a range is composed of many individual arrays. Nanoose Range, for example, has 24 arrays, while Dabob Bay has 7. The arrays are arranged in such a way that the array coverages overlap one another to provide continuous tracking on a target vehicle. Such overlapping areas are called crossover regions (see Figure 2), and produce two sets of track on the same target for the same time period.

Ideally, the corresponding points, or fixes described earlier, from each array in a crossover region should translate to identical points on the range coordinate system. In practice, this seldom happens.

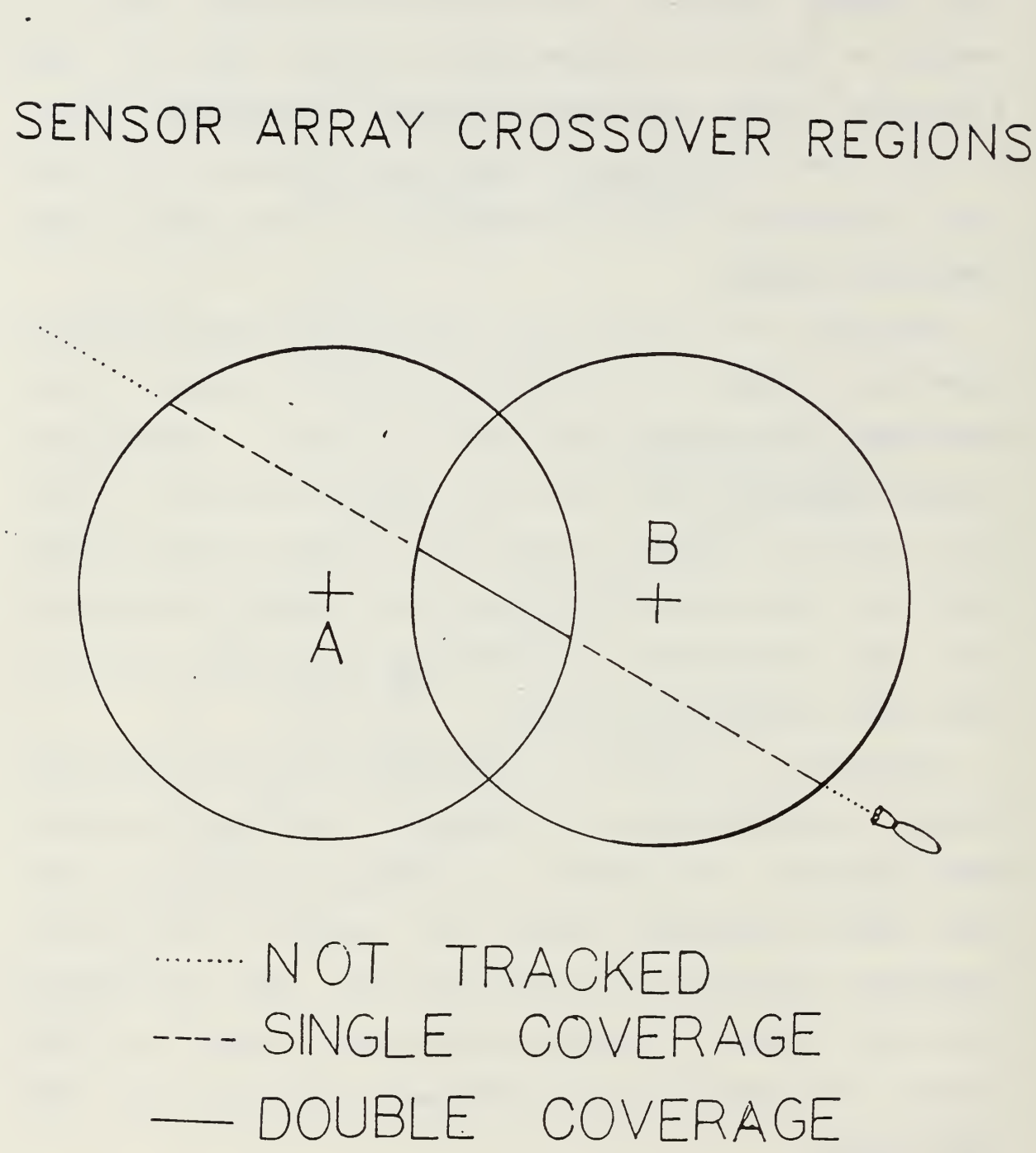


Figure 2 : Crossover Regions

Three major sources of this variability in crossover track data are:

(a) Slippage of the sensor arrays from their assumed positions in the range coordinate system.

(b) Time synchronization and instrumentation problems.

(c) Inhomogeneities and temporal variability in the water column.

Although the reasons for slippage of the arrays are speculative, the fact that slippage occurs is evidenced by the change in sensor locations after a range resurvey is performed. The question of discriminating timing errors, item (b), from slippage errors is on a future agenda. Investigation done by Main [Ref. 1] provides us with methodology for treating the random components of these errors. Item (c) falls into a broader category which may require extensive investigation. It may also provoke a reassessment of the assumed uniform horizontal quality of the range water column. It seems wise to account for slippage and instrumentation problems first. Our work is confined to (a).

The present research is in direct support of Professor Robert R. Read of the Naval Postgraduate School who has been working on the slippage question. He has produced a FORTRAN program (KEYMAIN) which takes the crossover data and estimates array position corrections using a non linear least squares algorithm. The surfaces that enter into this least squares optimization function are rather flat and the

values do not change much as the location corrections are varied. Replicated data is needed to quantify the extent to which the estimated corrections are within the range of natural variability. The simulation model in this thesis provides the tool by which this variability can be quantified. With the simulation, the needed replicated data can be quickly and easily generated, and scatterplots of displacement (change in location) versus rotation (change in orientation) can be made to investigate the inherent variability in the correction estimates.

III. DATA ANALYSIS

A. GENERATING RESIDUALS FROM REAL TRACK DATA

The real track data used for this study was taken from Nanoose range in September, 1982 (Figure 3). It was supplied as an N x 7 matrix, N representing the number of data points in the crossover data set and the columns representing the following:

Column 1	Point count. Every pulse emitted by the pinger is assigned a sequential number beginning at the start of the tracking run. Therefore, this column's value increases by one each row unless a data point is missing, which would be indicated by a sequential omission in this first column.
Columns 2-4	The X,Y, and Z coordinates of the target for the column one point count as determined by the first sensor in the crossover pair.
Columns 5-7	The X,Y, and Z coordinates of the target for the column one point count as determined by the second sensor in the crossover pair.

There were six such data sets used. These particular sets were chosen because each formed a straight line in the range space and so the best straight line path of the target could be estimated from the data. (In contrast, estimating the best curved path from curved track data would have been far more challenging, but with little or no gain in the examination of residuals.)

The idea was to take the track data, fit the best straight line possible through the data, then examine the

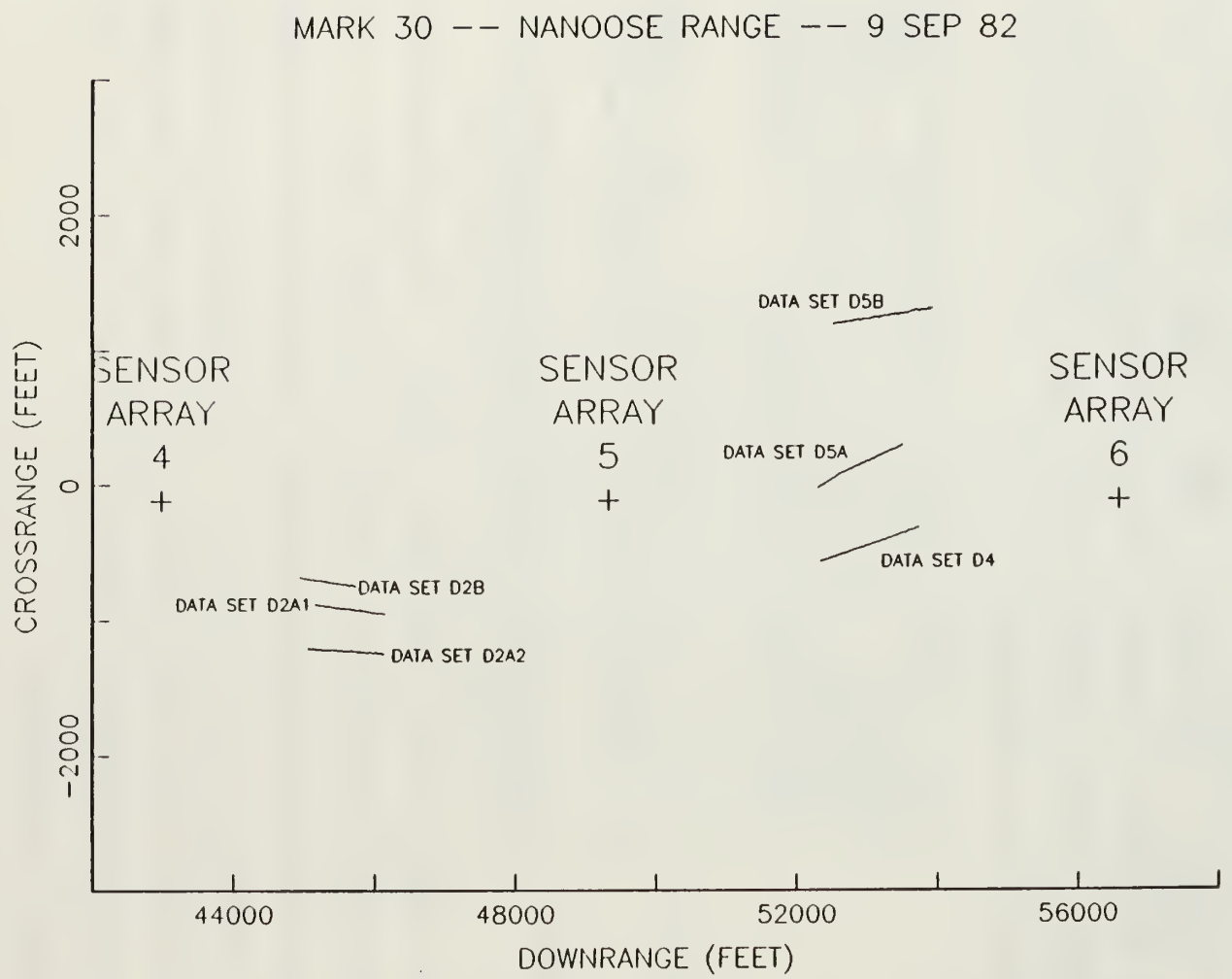


Figure 3 : Nanoose Range Data Diagram

residuals formed by subtracting the fitted straight line point sequences from the original data.

The commented FORTRAN program written to generate the residuals is included as Appendix A, and its mechanics are discussed below.

Since the data was given in a single matrix and residuals were desired for each sensor, the data was first separated into two tracks, each an N x 3 matrix; call them TR1 and TR2. Next, a 3 x 3 covariance matrix for each track was computed by the formula

$$COV = \left[\sum_{i=1}^{N} (TR(i)-\overline{TR})' \ (TR(i)-\overline{TR})\right] / N-1$$

where

TR(i) = 3 component row vector of the X,Y,Z coordinates of the i^{th} data point

$\overline{TR}$ = 3 component row vector of the average of the N (X,Y,Z) coordinates for that track data

N = number of data points

The best straight line estimate of the target path is the straight line such that the sum of the distances of each data point to the line is a minimum. This implies that the distance from each point to the line is a minimum - that is, each point should be projected onto the line orthogonally. The method employed to accomplish this orthogonal regression was that of principle components. Principle components requires that the eigenvector associated with the largest

eigenvalue of the covariance matrix be identified. It was computationally convenient to make a 3 x 3 matrix of the eigenvectors, with the first column being the eigenvector associated with the largest eigenvalue, the second column, the eigenvector associated with the second largest eigenvalue and third column, the eigenvector associated with the smallest eigenvalue. This done, the following matrix multiplication

$$\text{PROJECTION} = (\text{TR} - \overline{\text{AVE}}) \times \text{EIGENVECTORS}$$

where

TR = N x 3 track data matrix

$\overline{\text{AVE}}$ = N x 3 matrix made of N identical rows of the X,Y, and Z averages of the track data

EIGENVECTORS = 3 X 3 eigenvector matrix described above

yields the N x 3 matrix PROJECTION, whose entries are the orthogonal projections of the track data points onto the axes of a new 3 dimensional Cartesian coordinate system whose origin is located at $\overline{\text{TR}}$ (the (X,Y,Z) averages for the data set) and whose axes are rotated such that the new X axis is the best straight line that describes target path. For example, the first row of PROJECTION is a three component row vector; call the components (p1,p2,p3). Then the point (p1,0,0) is the projection of the first track data point onto the new X axis (known to be the best straight line that describes the target path), the point (0,p2,0) is the projection of the first track data point onto the new Y

axis, and the point (0,0,p3) is the projection of the first track data point onto the new Z axis. Since we are only interested in the projections onto the new X axis, we can replace the second and third columns of PROJECTION with zeros and have the coordinates of the best straight line path of the target. These coordinates are in the PROJECTION coordinate system, and must be translated back into the range coordinate system. Using the matrix PROJECTION (with last two columns = 0) and performing the following multiplication

$$OLD = EIGENVECTORS \times PROJECTION'$$

yields the 3 x N matrix of the track data points in the range coordinate system. Note that OLD must be transposed to get the N x 3 format desired. Since the mean had been subtracted off before computing the PROJECTION matrix, to get the data points back to the proper position requires that the means be added back in. The matrix addition

$$OLD' + \overline{AVE}$$

yields the N x 3 matrix of the orthogonal projections of the data onto the straight line path of the target in the original range coordinate system.

Actually, after having computed the PROJECTION matrix, there is yet another step to take before translating back into the range coordinate system. Based on an assumption of constant speed for the target, the track data points should be equally spaced since the pulses are emitted at regular,

precisely timed intervals. This could be accomplished by using an interval equal to the total distance divided by the number of data points, but if data points are missing in the track data set (and each set of track data was missing several points), then this method introduces an error. The method chosen should show that, if a single data point is missing, the gap between the consecutive points on the data track should be twice as long as if none were missing. The first column (the point counts) of the N x 7 track data matrix contains the information on missing points. Performing a simple least squares regression of the point counts onto the first column of the PROJECTION matrixtranslates this information into the best straight line point sequence for target motion. Now when the projections onto the first principle component are translated back onto the range coordinate system, the result is truly the best set of track points attainable using all the information contained in the track data set.

To obtain the residuals, the straight line estimated track points are subtracted, point for point, from the original track data. It was important to discover the distribution of the residuals for each track segment because to simulate the track segments later, residuals were simulated and then added to the straight line. Knowing the character of the residuals allowed the generation of very realistic track data for the simulation model.

B. ANALYZING TRACK DATA RESIDUALS

Each of the six track segments produced two sets of residuals - one set for each of the two sensors of the crossover pair. Each set was further divided into X, Y, and Z components. There were, therefore, 36 sets of residuals to be examined for distributional characteristics. Graphical analysis was performed on each set of residuals (histograms, cumulative density plots and QQ plots) and, when, from that analysis, a distribution for the residuals could be determined, formal statistical tests were performed to verify that the best distribution was chosen. The analysis showed the residuals to be normally distributed. The best and worst case graphical and analytical results are included in Appendix B. Note that there were some very good fits to a normal density (pp. 66-71) with statistical significances for the Chi Squared and Kolmogorov- Smirnov goodness of fit tests well above a very conservative .35 in all but one case. Even in the data that most poorly resembled a normal density (pp. 72-77), the Kolmogorov - Smirnov goodness of fit significance level never fell below .25. From this analysis it was concluded that a normal density for the residuals was accurate and appropriate for the simulation model.

C. TIME SERIES ANALYSIS

From a simulation point of view, it is very convenient to assume independence of the residuals between successive

points in the original track. If the assumption is not made, then one must resort either to using a point count interval of some integer value greater than one for making the simulated track from the original, or build an autoregressive model.

Concern for this correlation of the data over time led to a time series analysis of the residuals. This was done in two ways for each track from a single sensor. Recall that for each data point, residuals were generated in the X, Y, and Z directions of the range coordinate system. Each of these components was subjected to a time series analysis to determine whether there was a dependence in the errors in any single direction over time. Then, the distance of the data point from the straight line target path, given by

$$\text{SQRT}[\ (RES_x^2) + (RES_y^2) + (RES_z^2)\ ]$$

was examined to determine if the magnitude of the error of one point was correlated with the magnitude of the error of the next point. The time series analysis examines the dependence from point to point, on every other point, on every third point, and so on, so that the interval at which one can assume independence (indicated by an autocovariance value of zero) can be determined. Appendix C contains the autocorrelation graphs for the 6 data sets; first the error magnitudes are examined for both sensors of a data set, followed by the individual error analysis in the X, Y, and Z directions.

There was no clear cut interval for which independence seemed to emerge in the data sets. There is instead a random pattern of insignificant dependence from the beginning of the autocorrelation analysis, leading to the conclusion that point to point independence was appropriate for the simulation model. Attention is directed to the noise in the autocorrelation graphs with no distinct patterns emerging, and correlation magnitudes smaller than 0.2 in most cases. It was decided, therefore, that any important time correlation was not so large as to cause excessive or even noticeable error in the simulated tracks.

D. STUDY OF RESIDUALS

The study of the residuals yielded important interesting information, summarized in Table 1. The first column of Table 1 is the standard deviation of the residuals from the left sensor in the downrange (X), crossrange (Y) and depth (Z) directions, three values for each data set. Column 8 gives the corresponding information for the right sensor of the crossover data pair. Columns 2, 3 and 4 for each data set are the columns of a 3 x 3 correlation matrix for the residuals of the left sensor, columns 5, 6 and 7, corresponding information for the right sensor.

First, note the difference between the left and right sensor standard deviations of residuals in all three directions. Although there are some very good comparisons (data set D2A1, crossrange, and data set D5A, downrange)

TABLE 1
STANDARD DEVIATIONS OF RESIDUALS
AND
CORRELATION COEFFICIENTS OF RESIDUALS

SD LEFT	CORRELATION LEFT			CORRELATION RIGHT			SD RIGHT
			DATA SET D2A1 , N = 67				
0.842	1.000	0.916	-0.899	1.000	-0.863	0.787	0.721
2.141	0.916	1.000	-0.728	-0.863	1.000	-0.455	2.154
2.394	-0.899	-0.728	1.000	0.787	-0.455	1.000	2.063
			DATA SET D2A2 , N = 70				
1.484	1.000	0.859	-0.787	1.000	-0.360	0.040	1.163
2.396	0.859	1.000	-0.759	-0.360	1.000	-0.547	2.147
2.694	-0.787	-0.759	1.000	0.040	-0.547	1.000	1.976
			DATA SET D2B , N = 53				
0.693	1.000	0.788	-0.888	1.000	-0.949	0.902	0.937
2.000	0.788	1.000	-0.434	-0.949	1.000	-0.728	3.350
2.353	-0.888	-0.434	1.000	0.902	-0.728	1.000	3.042
			DATA SET D4 , N = 93				
0.497	1.000	0.741	-0.904	1.000	-0.681	0.380	0.502
1.955	0.741	1.000	-0.687	-0.681	1.000	-0.146	2.467
1.537	-0.904	-0.687	1.000	0.380	-0.146	1.000	1.848
			DATA SET D5A , N = 82				
1.781	1.000	-0.930	-0.429	1.000	-0.953	0.168	1.786
5.751	-0.930	1.000	0.152	-0.953	1.000	-0.148	6.645
1.338	-0.429	0.152	1.000	0.168	-0.148	1.000	1.878
			DATA SET D5B , N = 87				
0.886	1.000	-0.408	-0.450	1.000	0.445	0.395	0.717
2.931	-0.408	1.000	-0.454	0.445	1.000	-0.279	1.613
3.397	-0.450	-0.454	1.000	0.395	-0.279	1.000	4.254

there are also some wide disparities (data set D2B, crossrange and data set D5B, crossrange). Using the variance ratio test described in Larson [Ref. 2: pp. 449], a statistical test was performed to determine whether the variances (the recorded standard deviations, squared) of the left and right sensor arrays for downrange, crossrange and depth were the same for a given data set. The results are given in Table 2. Using 0.05 as the basis for rejection of the null hypothesis (H_0: the two variances are the same), 8 of the 18, or 44%, of the individual tests fail. With a confidence level of .95, one would expect roughly 1 failure in the 18 tests. Eight failures is strong evidence that the standard deviation figures do not match very well.

Recalling the test data from Figure 3 (p. 18), it is seen that there are three sensor arrays that contributed the data: arrays 4, 5 and 6. In data sets D2A1, D2A2 and D2B, array 4 is the left array and array 5 is the right sensor array. For data sets D4, D5A and D5B, sensor array 5 is the left array and array 6 is the right array. One might expect that the standard deviations in any single direction would be the same for a given sensor. This turned out not to be the case. Using Bartlett's test for the equality of several variances [Ref. 3: pp. 225-227], the hypothesis that the variances for a given sensor in a given direction are equal was tested. The results are given in Table 3. The first two sections of the table compare the 3 standard deviation

TABLE 2 TEST OF THE HYPOTHESIS THAT THE STANDARD DEVIATIONS WITHIN A DATA SET ARE EQUAL

	VARIANCE RATIO	DEGREES OF FREEDOM	LEVEL OF SIGNIFICANCE
DATA SET D2A1			
DOWNRANGE	1.363	66	.211
CROSSRANGE	.988	66	.935
DEPTH	1.363	66	.229
DATA SET D2A2			
DOWNRANGE	1.627	69	.045
CROSSRANGE	1.244	69	.364
DEPTH	1.858	69	.011
DATA SET D2B			
DOWNRANGE	.547	52	.031
CROSSRANGE	.357	52	.0003
DEPTH	.598	52	.067
DATA SET D4			
DOWNRANGE	.982	92	.884
CROSSRANGE	.628	92	.027
DEPTH	.692	92	.079
DATA SET D5A			
DOWNRANGE	.995	81	.939
CROSSRANGE	.749	81	.195
DEPTH	.508	81	.003

TABLE 2 (continued)

DATA SET D5B

DOWNRANGE	1.527	86	.051
CROSSRANGE	3.300	86	0.0
DEPTH	.638	86	.038

Table 3 TEST OF THE HYPOTHESIS THAT THE STANDARD DEVIATIONS OF RESIDUALS OF A PARTICULAR SENSOR ARE EQUAL

Chi Squared Random Variables

DEGREES OF FREEDOM	.99 QUANTILE	.999 QUANTILE
2	9.21	13.82
5	15.09	20.52

	DOWN RANGE	CROSS RANGE	DEPTH	DEGREES OF FREEDOM
SENSOR 4 (LEFT)	39.57	2.02	1.41	2
SENSOR 5 (RIGHT)	14.75	16.25	13.96	2
SENSOR 5 (LEFT)	131.14	101.33	91.40	2
SENSOR 6 (RIGHT)	148.72	175.18	84.31	2
SENSOR 5 (LEFT AND RIGHT)	153.04	150.87	105.51	5
LEFT SENSORS	171.45	162.29	94.48	5
RIGHT SENSORS	168.37	236.97	107.15	5

values for each sensor in a single direction. The test statistic is distributed as a Chi Squared random variable with 2 degrees of freedom. The .99 and .999 quantiles for such a random variable are 9.21 and 13.82. In only two cases of the 12 trials would the 3 standard deviations be considered statistically the same.

The third section of Table 3 is included because sensor array 5 was used as both a right and left array. Therefore, there were actually six values for downrange, crossrange and depth for that particular array. Using the same test as before, the question of whether the six values were statistically the same was investigated. The tabulated statistic for the 3 cases is distributed as a Chi Squared random variable with 5 degrees of freedom. The .99 and .999 quantiles for such a ramdom variable are 15.09 and 20.52. In every case, the p-value of the test is essentially equal to 0.0. The results are highly significant.

Finally, the bottom of the table investigates whether downrange, crossrange and depth residual standard deviations are the same for left and right sensors in general. Employing the same test for the 6 values produced the tabulated statistics. Since the test statistic is again Chi Squared with 5 degrees of freedom, the level of significance in every case is essentially 0.0.

A third question of interest from Table 1 is whether the correlation coefficients are the same for the left and

right sensors in a data set. This was investigated in the following manner.

First, the normalizing, inverse hyperbolic tangent transformation [Ref. 3: p. 365] was applied to each of the off diagonal correlation coefficeients in the correlation matrices. (Note that the matrices are symmetric, so there are only three values of concern for each matrix.) This transformation makes each of the values normal with mean

$$\frac{1}{2} \ln \frac{1 + rho}{1 - rho}$$

and variance

$$\frac{1}{N - 3}$$

where N is the number of data points contributing to the correlation. Under the assumption that the two independent sample correlation coefficients come from the same population, their difference is distributed normally with mean 0 and variance

$$\frac{2}{N - 3}$$

We can therefore go to the standard normal tables to obtain the significance of the statistic

$$\frac{Z_1 - Z_2}{\text{SQRT } [2/(N-3)]}$$

which tests the hypothesis that the two correlation coefficients are equal. These values are tabulated in Table

TABLE 4 TEST OF THE HYPOTHESIS THAT THE CORRELATION COEFFICIENTS WITHIN A DATA SET ARE EQUAL

DATA SET	CORRELATION COEFFICIENT	Z1 - Z2	STANDARDIZED NORMAL	LEVEL OF SIGNIFICANCE
D2A1	X,Y	2.867	16.227	0.0
D21A	X,Z	2.534	-14.333	0.0
D2A1	Y,Z	-.434	-2.453	0.014
D2A2	X,Y	1.668	9.654	0.0
D2A2	X,Z	1.103	-6.384	0.0
D2A2	Y,Z	-.381	-2.203	0.028
D2B	X,Y	2.888	14.438	0.0
D2B	X,Z	-2.896	-14.481	0.0
D2B	Y,Z	.459	2.293	0.022
D4	X,Y	1.783	11.959	0.0
D4	X,Z	-1.894	-12.706	0.0
D4	Y,Z	-.695	-4.659	0.0
D5A	X,Y	.206	1.293	0.196
D5A	X,Z	-.628	-3.948	0.0
D5A	Y,Z	.302	1.900	0.057
D5B	X,Y	-.911	-5.904	0.0
D5B	X,Z	-.903	-5.850	0.0
D5B	Y,Z	-.203	-1.443	0.149

4. To get a significance of greater than .05 in this test requires a value between +/- 1.96, so only 3 of the 18 pairs of correlation coefficients are statistically equal, giving very strong evidence that the correlation coefficients of the residuals as recorded by the two sensors in a crossover pair are, in general, not equal.

The practical significance of the preceding tests is that there appears to be much local variation in the range and that one single model for simulating random replications of underwater track is not apparent. Therefore, each case must be simulated and studied separately, and the study of the error estimates distributions done on a case by case basis.

IV. SIMULATION MODEL

The simulation model itself is a logical extension of the residual generation program. In fact, the method used to simulate a data track was to simulate residuals and add them to the fitted straight line obtained for that track, rather than to start from scratch and simulate a totally new track at each iteration. This method was very quick and yielded very good simulated track segments. Figure 4 on the following page is graphical comparison of a typical original track segment overlaid by a track segment simulated by the method stated above. The comparison demonstrates the realistic quality of the simulated track.

The regression method used to compute residuals of a track segment also produced the best straight line in three dimensional space to approximate the target path through the water. Given the straight line, the residuals themselves, and the assumption of normality of the residuals, it is possible to simulate a set of residuals from normal (0,1) deviates and add them to the straight line to obtain a simulated track.

In simulating residuals, the object is to compute an N x 3 matrix, using normal (0,1) deviates, whose vectors of X, Y, and Z components are normally distributed with mean 0 and whose covariance matrix is equal to the covariance

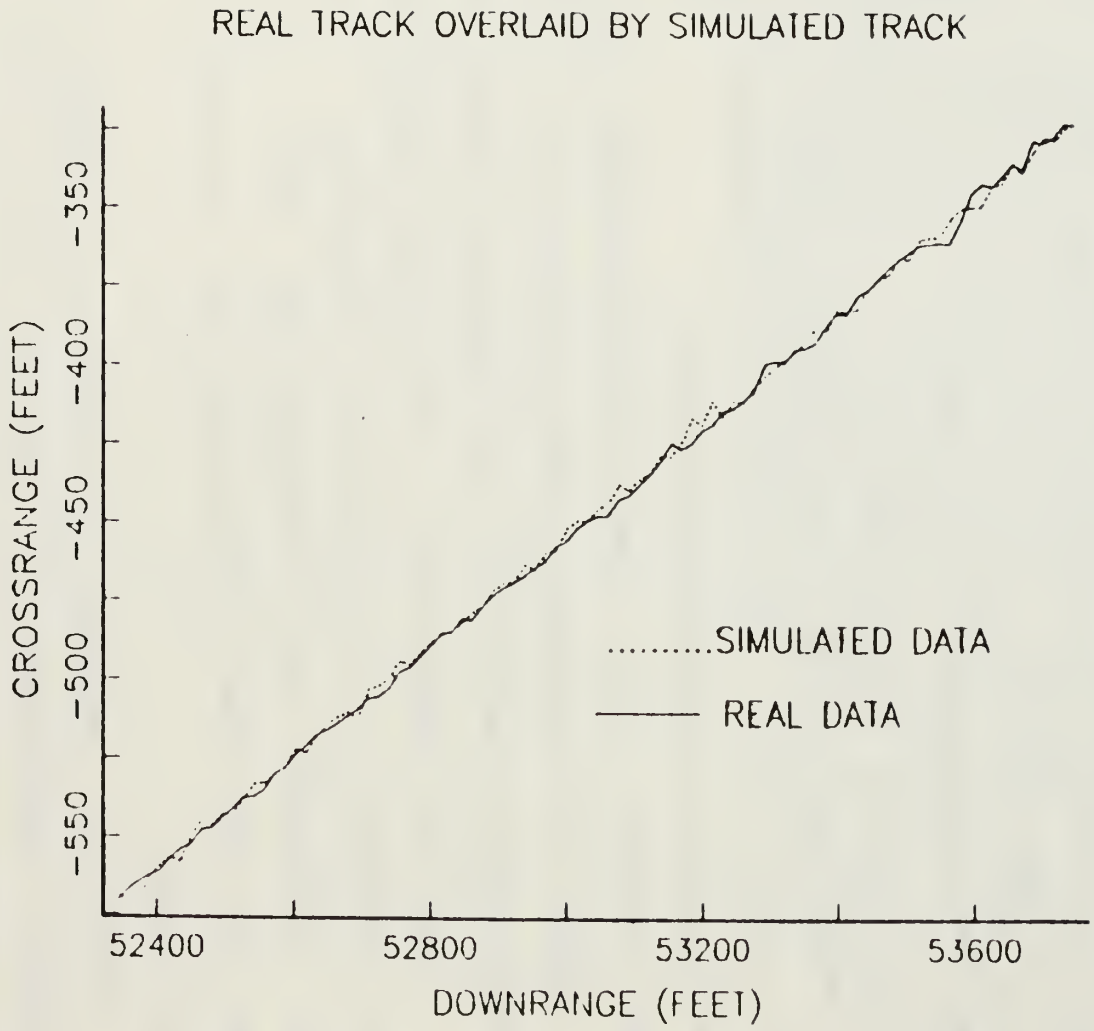


Figure 4 : Comparison of Original and Simulated Track Data

matrix of residuals. That is

$$R = T \times X'$$

where

R = 3 x N matrix of simulated residuals

X = N x 3 matrix of N(0,1) deviates

T = 3 x 3 transformation matrix such that

T x T' = covariance matrix of residuals.

Note that R must be transposed to get the desired N x 3 residual format.

It is easy to show that any 3 x 3 matrix T will not alter the mean of 0.

$$E[R] = E[T \times X']$$

$$= T \times E[X']$$

Since the expectation of X -- consisting of normal (0,1) deviates -- is identically 0,

$$E[R] = T \times (0) = 0$$

giving the desired result.

The condition that T x T' is equal to the covariance matrix of the residuals is necessary because

$$COV[R] = E[R \times R']/N \quad \text{(the mean is 0)}$$

$$= E[T \times X' \times X \times T']/N$$

$$= T \times E[X' \times X]/N \times T'$$

(since T is a linear operator)

$$= T \times I \times T'$$

(because the covariance matrix of N(0,1) random variables is the identity matrix)

$$= T \times T'$$

The problem now becomes one of finding a 3 x 3 matrix such that

$$COV[R] = T \times T'$$

We have the covariance matrix of the residuals. Let T be an upper triangular matrix. Then

$$\begin{bmatrix} T_{11} & T_{12} & T_{13} \\ 0 & T_{22} & T_{23} \\ 0 & 0 & T_{33} \end{bmatrix} \times \begin{bmatrix} T_{11} & 0 & 0 \\ T_{12} & T_{22} & 0 \\ T_{13} & T_{23} & T_{33} \end{bmatrix} = \begin{bmatrix} COV_{11} & COV_{12} & COV_{13} \\ COV_{21} & COV_{22} & COV_{23} \\ COV_{31} & COV_{32} & COV_{33} \end{bmatrix}$$

Noting that the covariance matrix is symmetric and performing the matrix multiplication yields

$$T_{33} = SQRT\ (COV_{33})$$

$$T_{23} = \frac{COV_{32}}{T_{33}}$$

$$T_{22} = SQRT\ (COV_{22} - T_{23}^{2})$$

$$T_{13} = \frac{COV_{13}}{T_{33}}$$

$$T_{12} = \frac{COV_{21} - T_{23}T_{13}}{T_{22}}$$

$$T_{11} = SQRT(COV_{11} - T_{13}^{2} - T_{12}^{2})$$

Since the matrix T is easily computed and X can be generated from a random number generating package, the residuals, R, are quickly and economically computed for as many simulated tracks as desired.

Adding the residuals thus formed to the best straight line target path of the original data yields a simulated track.

The commented FORTRAN program written to perform the simulation is included as Appendix A. It is basically a driver program that generates a user specified number of simulated tracks and interfaces with KEYMAIN, which estimates displacement and angular rotation values. Because some of the user friendly attributes of KEYMAIN interfere with the speedy generation of the simulation's required output parameters, the package has been altered somewhat to increase speed. The output of the driver program is two data files. One data file, ROTATE.DAT, is an N x 4 matrix that gives, in the first column, the maximum angle of rotation of the sensor, and, in the last 3 columns, provides the ordered Euler angle components that form the single maximum rotation angle. The second file, DISPLACE.DAT, is also an N x 4 matrix. The first column is the magnitude of displacement of the array, while the last 3 columns give the X, Y, and Z axis components of displacement.

V. INTERFACE WITH EXISTING PROGRAMS

The software developed to compute residuals and generate simulated track segments are the original work of the author, aided by such canned routines as eigensystem analysis, random number generators and vector arithmetic. These canned subroutines came from the IMSL Library of Mathematic and Statistical functions developed for the IBM PC computers [Ref.4], and are compiled in machine language libraries not reproducible here. The author's FORTRAN code is listed as Appendix A.

In order to produce the estimates of sensor displacement and rotation, however, it was necessary to interface with a large stand alone FORTRAN package, KEYMAIN. This program takes as input crossover region data sets (real or simulated) and produces the estimates of displacement and rotation of the second sensor of the crossover pair, based on the assumed accurate position of the first. The orientation correction output by KEYMAIN is actually a three valued vector of ordered angular rotations in the XY, XZ and YZ planes. Similarly, the location correction is a three valued vector of displacements in the X, Y and Z directions. To reduce complexity, the ordered Euler angles were reduced to a single maximum angle of rotation about an appropriately tilted axis, and the three components of displacement were

reduced to a single quantity, magnitude of displacement. Thus the six dimensional quality of position corrections was reduced to two.

KEYMAIN was designed as a stand alone product and as such is very user friendly. It was also designed to take not just one, but several, crossover data sets and produce displacement and rotation estimates for several sensors at once. Because its use in the simulation was to process a single simulated crossover data set, and because it was being called as a subroutine rather than used as a stand alone package, significant changes were required in the program to obtain fast simulated results uninterrupted by the now unnecessary user friendliness. Not only did this require the modification of the executive driver routine, but also modification of several of the called subroutines.

VI. RESULTS

It is imperative that the simulated track segments "look" and "act" like the original track segment they are simulating. That the simulated track segments act like the original is guaranteed by the equations: the residuals will have mean zero by definition and their covariance matrices were computed to be the same as the original's. For visual verification, Appendix D is included. Appendix D contains plots of each original track segment overlaid by one of the tracks simulated from it. A good fit is apparent in all cases.

The simulation model can produce the data needed to produce a scatterplot of magnitude of displacement (in feet) versus maximum angle of rotation (in radians) like the schematic shown in Figure 5. This represents an imaginary situation where a target vehicle was driven through a single crossover region on a range 700 times over the exact same track, and the results were fed into the program to produce the displacement and rotation values. It acts as a data base, or sampling distribution, for an array whose position has not changed, and the graph depicts the natural variability of the data. The contours represent some theoretical confidence levels for that particular sensor. For example, take the outermost contour, labeled .90. In a

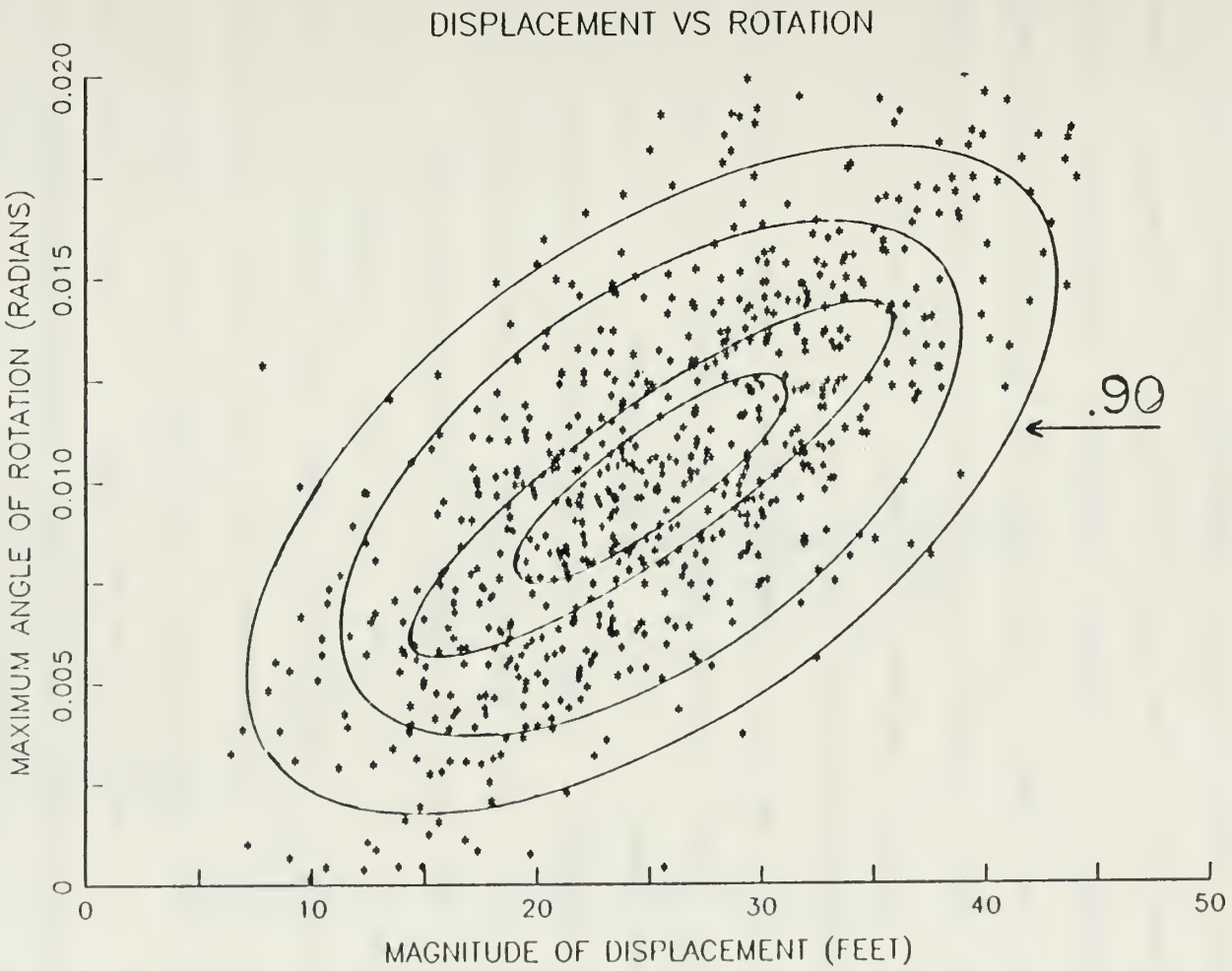


Figure 5 : Demonstration Schematic of Displacement vs Rotation

future tracking exercise, crossover data can be fed into the program that produces displacement and rotation estimates. That will produce a point on the graph. If that point lies outside the contour, we have good evidence that the sensor has moved. Statistically speaking, one would have a 10 percent chance of rejecting a true null hypothesis, given a null hypothesis of no sensor movement. Conversely, if the point plotted closer to the middle of the graph, there would be insufficient evidence to support the hypothesis of sensor movement, and the apparent movement would be attributed to the inherent variability in the data.

It is unreasonable to expect that a target vehicle could be driven along the same track 700 times, nor would the range operators be likely to attempt it. The simulation program, however, needs only one straight line segment of track through a crossover region, and can then generate as many track data sets as needed to produce the graph.

Figures 6 through 11 are plots produced from the simulation model using the data sets from Figure 3 (p. 18). Figure 9 is a "well-behaved" plot that could conceivably, with many more runs, yield the type of graph displayed in Figure 5. In fact, the points appear so evenly distributed that one could conceivably assume independence between the rotation and displacement values.

Although computationally attractive, independence is thought to be a poor assumption, because if a sensor has

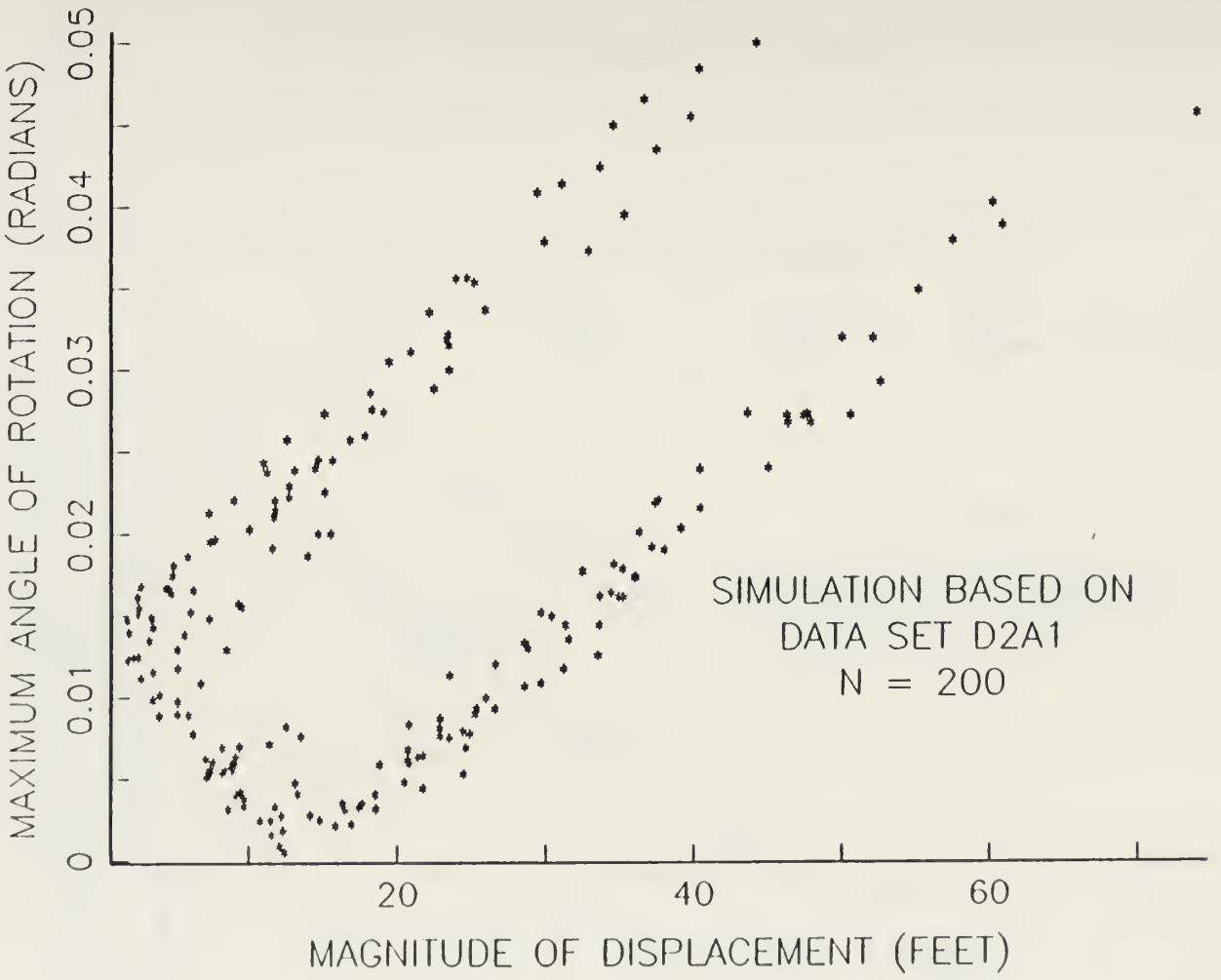


Figure 6 : Displacement vs Rotation Data Set D2A1

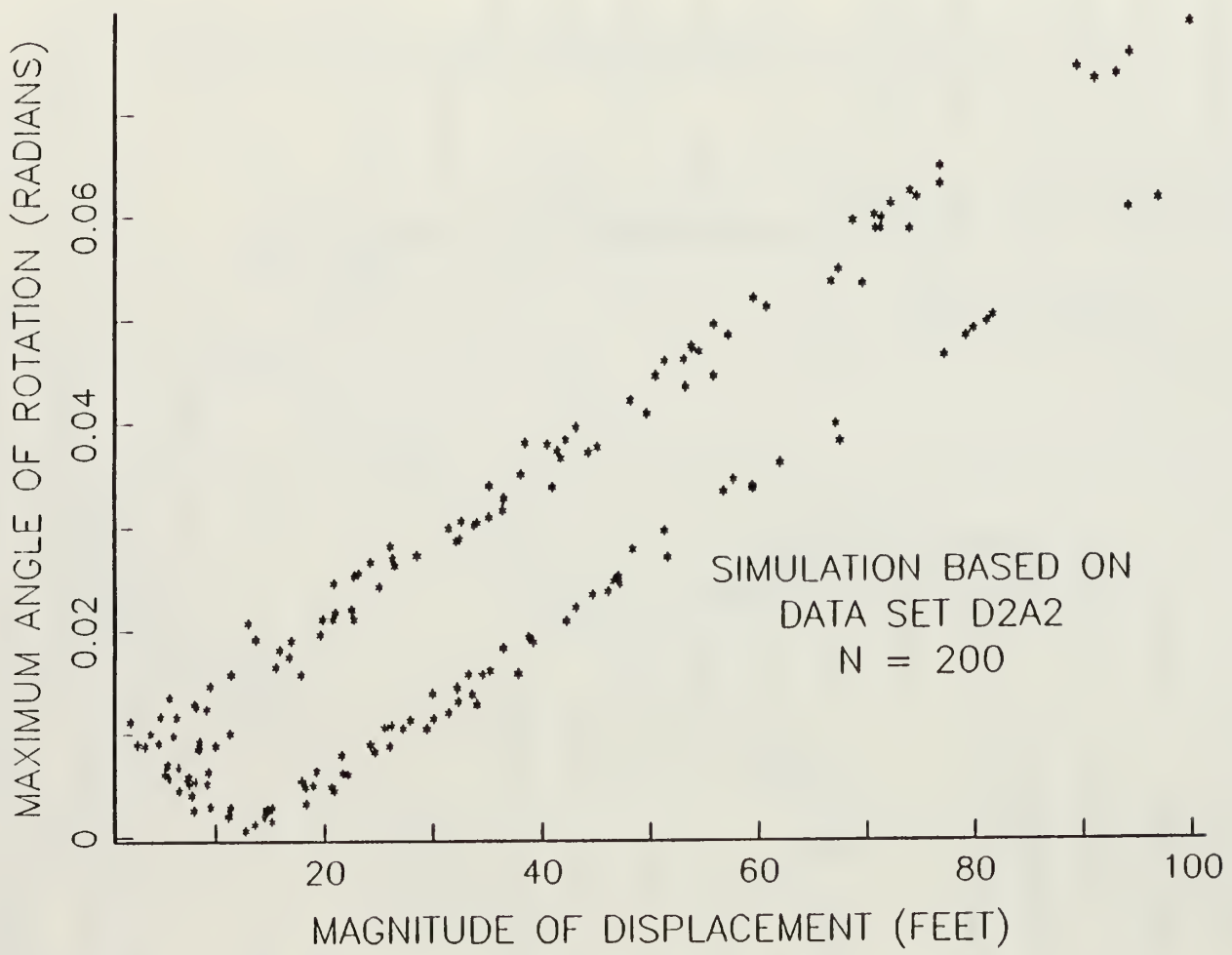


Figure 7 : Displacement vs Rotation Data Set D2A2

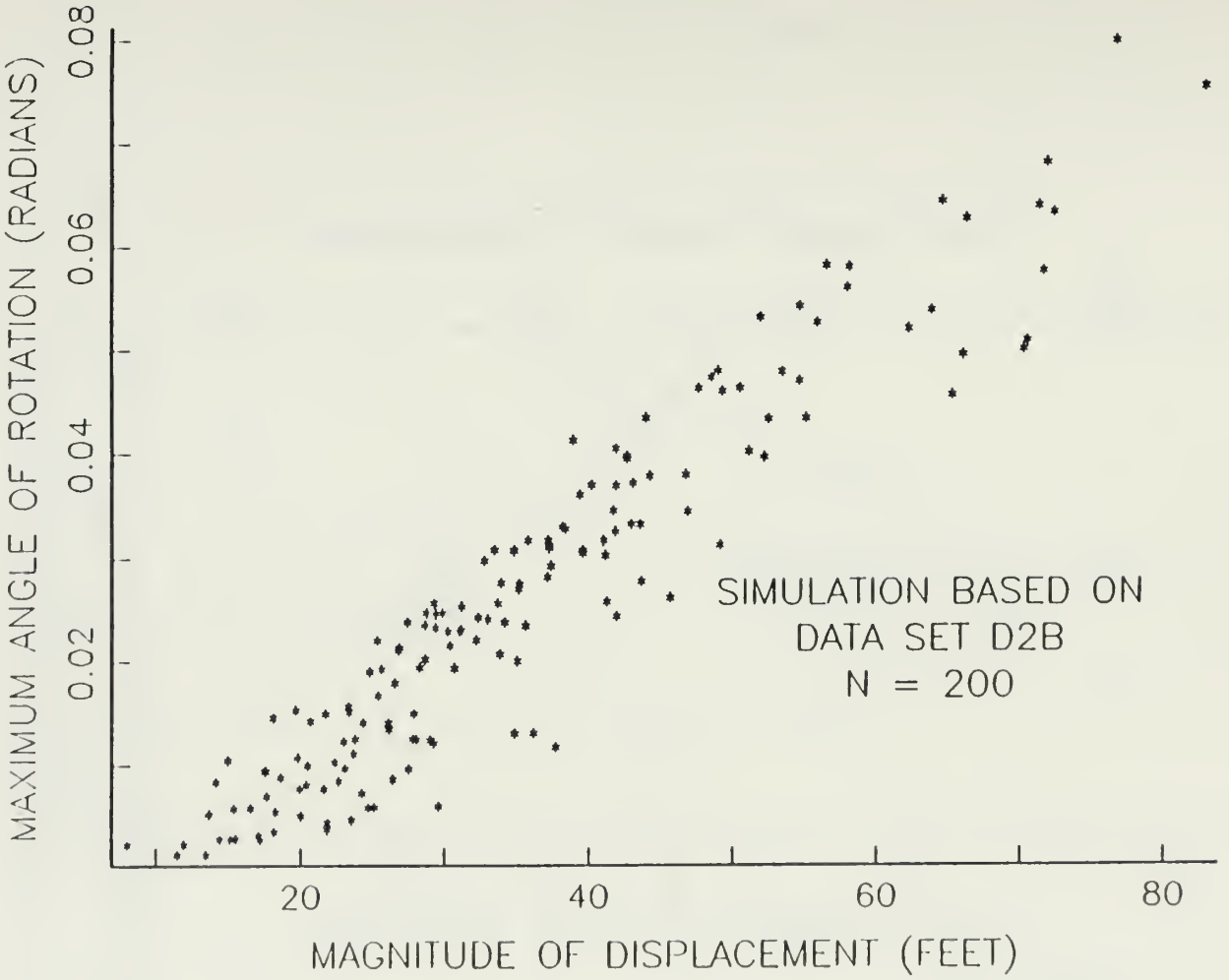


Figure 8 : Displacement vs Rotation Data Set D2B

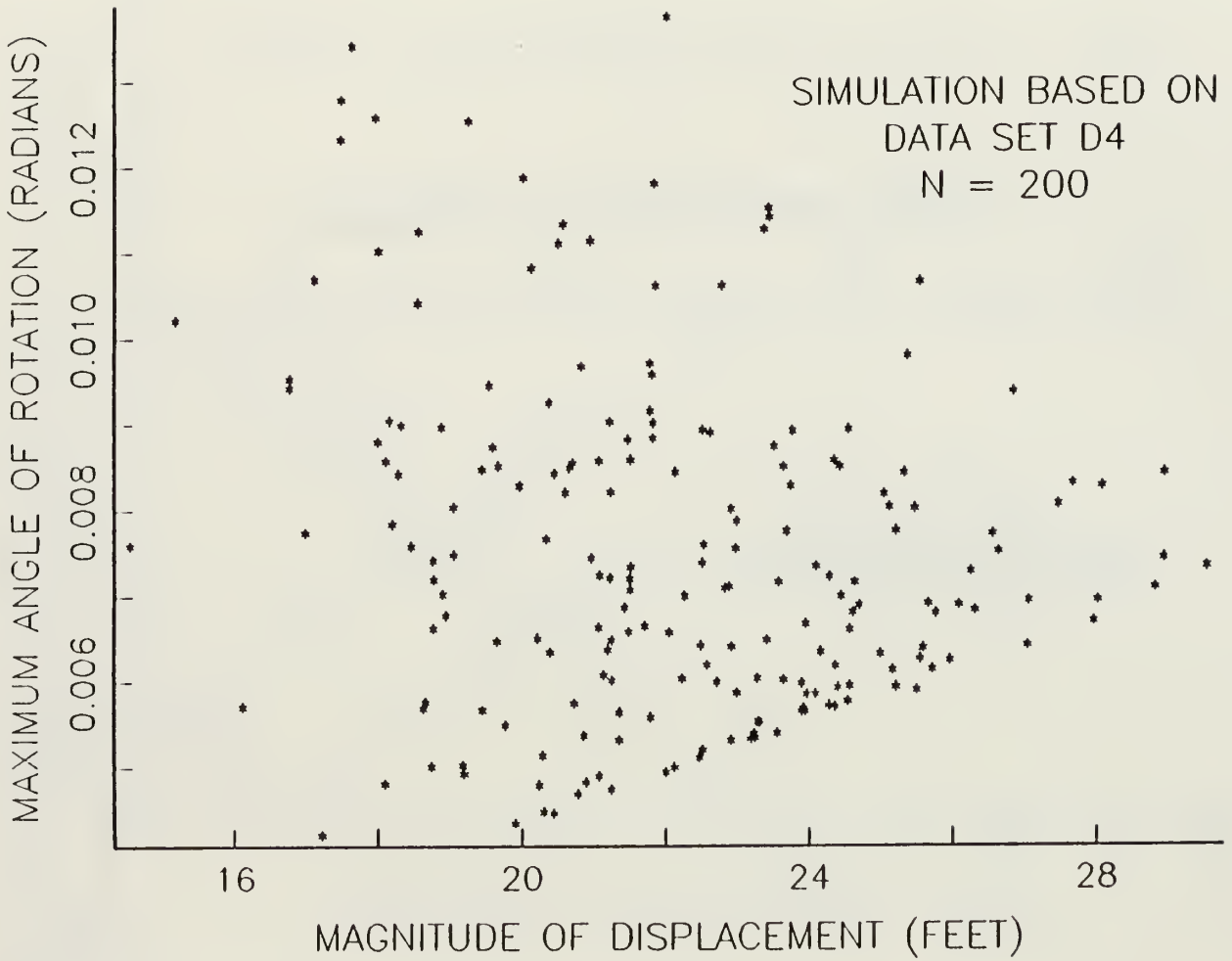


Figure 9 : Displacement vs Rotation Data Set D4

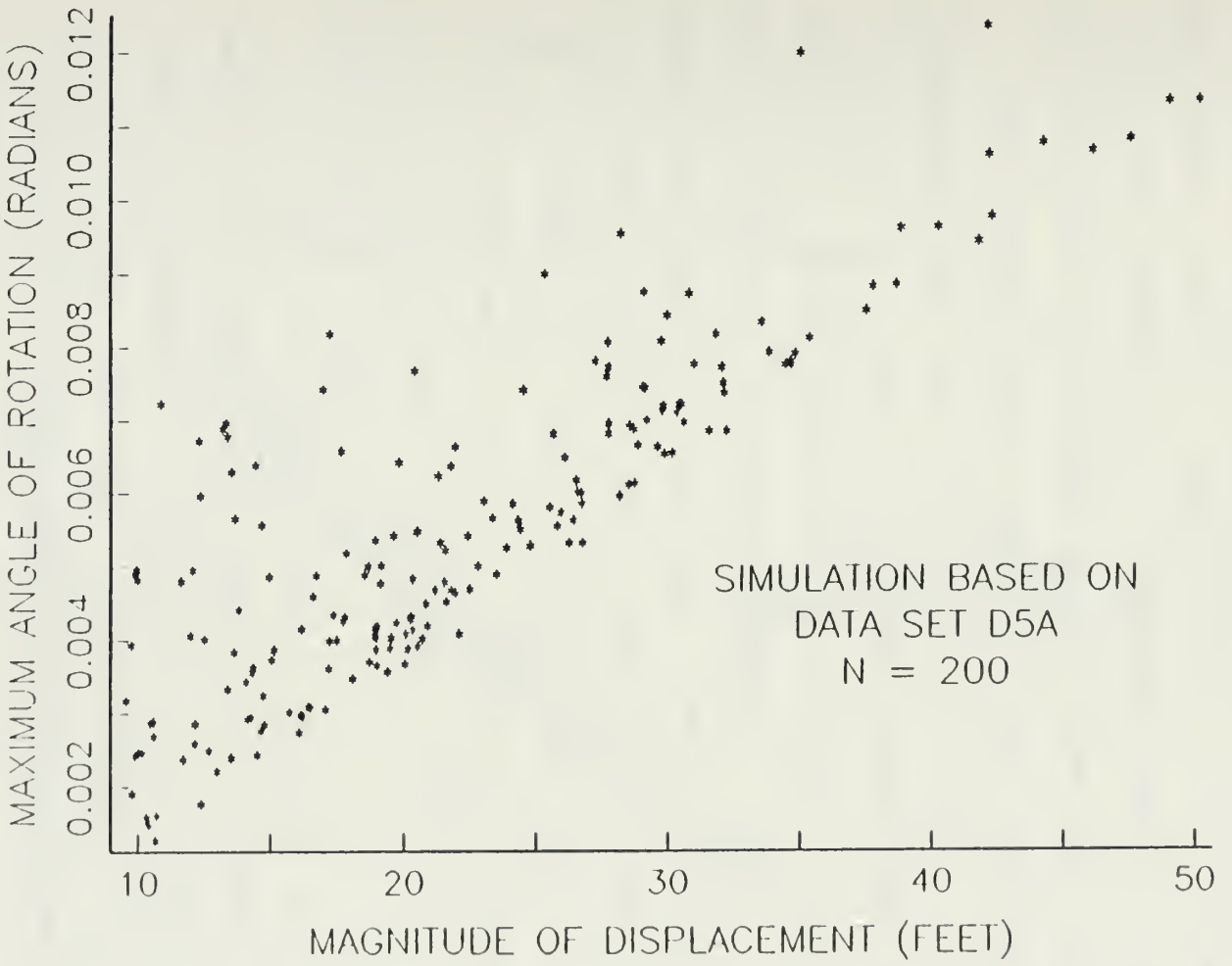


Figure 10 - Displacement vs Rotation Data Set D5A

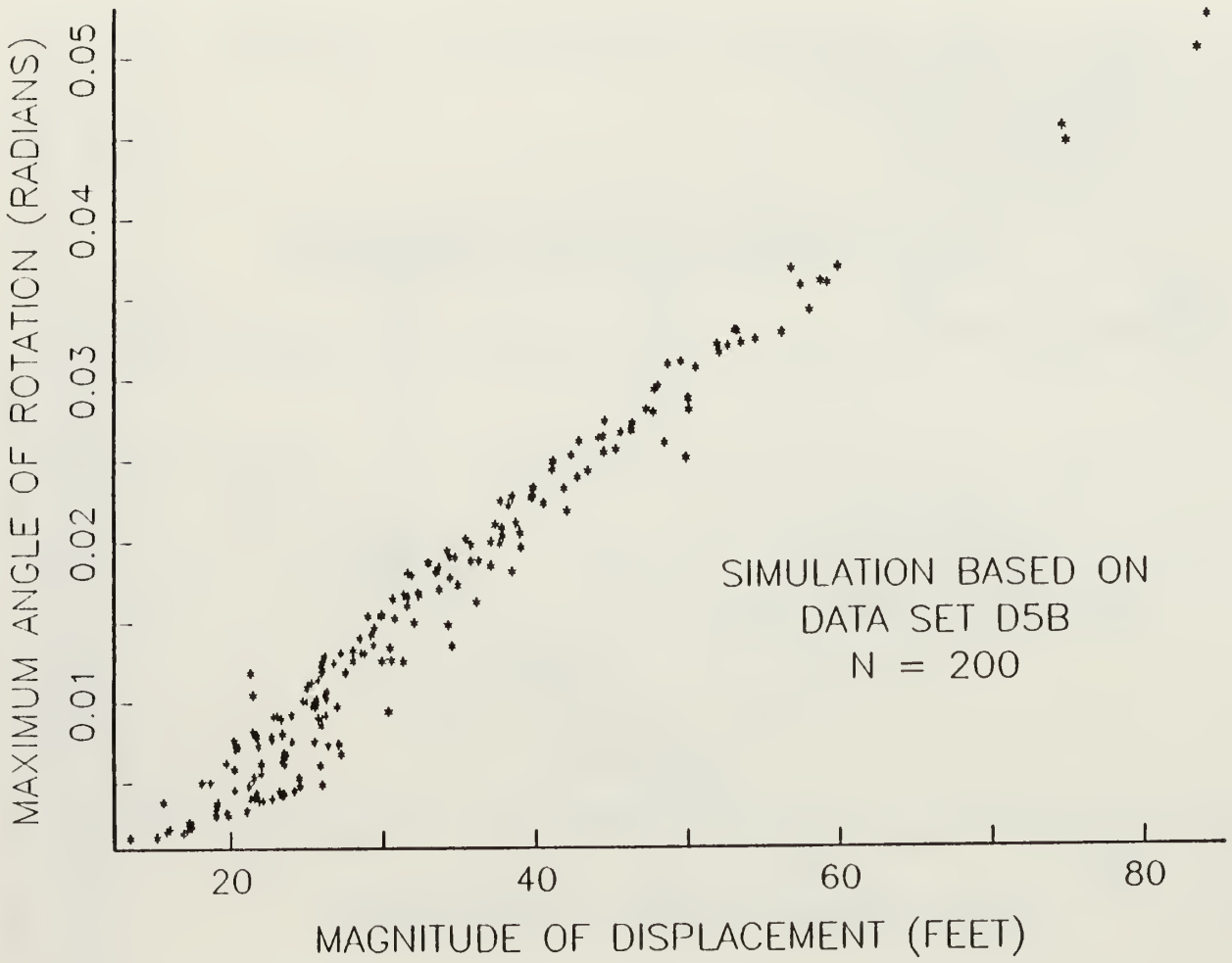


Figure 11 : Displacement vs Rotation Data Set D5B

displaced a great distance, it has also had ample opportunity to rotate. This observation is speculative, but seems to be borne out in the remaining figures where a strong positive correlation appears to exist between the displacement and rotation values. The fishhook appearance on these graphs is a result of making the correction estimates positive regardless of the direction of displacement or rotation. The graphs taken as a whole serve to illustrate the variable nature of the track data.

These three graphs, drawn from the simulation, vividly illustrate the need for further study. Two points concerning the data on which these graphs were based are perhaps of some importance and may begin to explain the strange nature of the graphs produced. First, the data is fairly old, taken in September, 1982. The range operators from NUWES state that their capabilities have improved in the interim 3 years and that newer data could prove significantly more accurate. Second, the data was taken on a single day, from a single range, using only 3 of the 24 sensors on the range.

The great variety of graphs produced points to the necessity to do much more research and to the utility of a simulation model to accomplish it. In addition to the aforementioned use of newer data, the following considerations warrant study to ascertain their possible effects:

(a) Day to day variations on the range -- It is not clear at this time whether the character of the water column in which the target operates is invariant over time. Differences in salinity and/or temperature could conceivably evolve over time, affecting the accuracy of the track data.

(b) Seasonal variations on the range -- While it is not clear whether changes occur on the range on a daily basis, it certainly seems reasonable to expect a variation from season to season. Our data, taken from a single range on a single day, was insufficient to explore this.

(c) Location on the range-- Current practice on a tracking day is to take one sample of the water column on the range, checking for salinity, temperature, and other factors that affect the sound velocity profile, and assume that the results hold true for the duration of the exercise for the entirety of the range. The possibility of a daily change was discussed earlier. Here we mention the possibility of different water characteristics from one end of the range to the other.

(d) Geometry of tracking runs -- It is possible that varying the geometry of the tracking run could result in changes to the quality of the data recorded. It can be seen from figure 3 (on page 18) that all of the data used for the simulations was from tracks that run predominantly downrange with relatively little crossrange change and virtually no depth change. (Although depth cannot be seen from figure 3, it was examined for all the tracks.)

(e) Depth of target -- Depth appears to be the least reliable of the three dimensions recorded during tracking runs. Our data is from targets that operated in a narrow depth band. Deeper or shallower targets could significantly affect the data.

(f) Inhomogeneities in the water -- It is quite conceivable that undetected inhomogeneities in the water could significantly affect the quality of the data. It would appear that currents and turbulence could affect the passage of sound through the water column, but quantifying these disturbances could be a major practical problem.

Precisely characterizing the variability of the correction estimates is elusive and the preceding considerations invite much future research before that goal is reached. The existence of this simulation model brightens the outlook for ultimate success.

VII. AREAS FOR FUTURE WORK

Problems encountered in the development of the simulation model and intuition gained as a result of working on it gave birth to several areas for potential future study. Some of these are listed below.

A. RESIDUALS FOR CURVED CROSSOVER DATA TRACKS

It was convenient in this simulation model to use straight line segments of track to get residuals. This is because the method of principle components works only for straight lines in N-space, rather than for curves. If a method could be developed to regress the data for any track onto its best path, regardless of curvature, one major obstacle in the use of the simulation would be removed. Whereas now we are limited in the type of data we can use, such a method would enable the use of all crossover data.

B. WEIGHTED DATA BASED ON DISTANCE TO SENSOR ARRAY

It was assumed in the model that recorded track data was uniformly accurate and reliable, regardless of the distance from target to sensor. That is, no distinction was made between the accuracy of the data when a target vehicle was "close" to a sensor and when the target was far away. Some sort of weighting scheme may prove beneficial and help smooth out the current data discrepancies.

C. TRISENSOR CROSSOVER DATA

Although smaller in size, there exist areas on the range where a target may be tracked simultaneously by three sensors at once. These trisensor crossover regions could provide insight into the accuracy of the position of a sensor and may yield more accurate estimates of displacement and rotation.

D. TRACK DATA SELECTION PROCEDURES

It is not now known whether a greater number of data points in a track segment yields better results in the simulation and intuition is limited on this point. Research in this area could provide valuable guidelines for selection of data to use in the simulation in the future.

E. DECISION RULES TO DETERMINE SENSOR MOVEMENT

After accurate characterization of the variability of track data and correction estimates is made, the next logical and extremely useful step would be the construction of statistically sound decision rules to determine the following:

(a) The discrepancy noted between the tracks in a crossover data set can be explained by the inherent variability of the data.

(b) The discrepancy noted between the tracks in a crossover data set can be quantified by the sensor slippage model and can be computationally corrected.

(c) The discrepancy noted between the tracks in a crossover data set cannot be explained by the model of sensor movement and some other explanation must be sought.

Clearly, the last option is the least desirable, but if that situation is present, it needs to be noted.

F. COMPUTATIONAL RANGE RESURVEY

If the method of providing correction estimates can be proven to be reliable and accurate, it provides a potential method of range resurvey that has several advantages over the current method. First, the range would not need to be shut down to resurvey. In fact, range use would be mandatory to keep up to date sensor array positions. Second, it would be less expensive than the current method, replacing the equipment and manpower intensive current process with relatively inexpensive computer assets. Third, it would take less time. Resurvey of a single sensor array on the range can take up to a day; generating corrections from track data takes seconds. Fourth, since the current method uses a craft on the surface equipped with a pinger, all the pings must travel through the first 150-200 feet of the water column, where the sound velocity profile is quite variable and most difficult to determine, to get to the sensor array. In contrast, the underwater target vehicles tracked by the arrays are typically in 400-600 feet of water where the sound velocity profile is much smoother and easier to predict. Thus, one source of variability in determining sensor position is reduced.

It is not envisioned that this computer method could ever replace the current survey process, but rather augment

it. Some way to determine the position of one sensor is necessary before KEYMAIN can even begin to function. However, if this computer process could augment current resurvey efforts, or reduce the frequency with which resurveys must be made, significant savings in time and money could result.

APPENDIX A

FORTRAN LISTING FOR PROGRAM SIMDAT

```
      PROGRAM SIMDAT2
C
C...This program simulates 3-D track data based on a
C...specific real track specified by the user.
C...User inputs are:
C...      1.  track segment data file to be simulated
C...      2.  number of the left and right sensor in the
C...          crossover pair
C...      3.  number of simulated tracks desired
C...      4.  request for sample simulated track (YES or NO)
C...      5.  random number generator seed
C...The program output is:
C...      1.  file of residuals from the original track
C...          (RESIDUAL.DAT)
C...      2.  file of a simulated crossover track, if
C...          requested (SIMTRACK.DAT)
C...      3.  file of displacement values (in feet) for the
C...          right sensor of each simulated track in 4
C...          columns
C...              Col 1    magnitude of displacement
C...              Col 2-4  X,Y,Z components of displacement
C...      4.  file of rotation values (in radians) for the
C...          right sensor of each simulated track in 4
C...          columns
C...              Col 1    maximum angle of rotation
C...              Col 2-4  ordered Euler angles of rotation
C
C...VARIABLE DECLARATION
C
       INTEGER*4 N, I, J, K, IER, BIG1(3), BIG2(3), SIMS,
     +  POINT(130), TRACKS, IDL, IDR, TRKOUT
C
       CHARACTER DSNAME*13
C
       REAL*4 NORM(260)
C
       REAL*8 TRACK(130,6), TR1(130,3), TR2(130,3), MBAR1(3),
     +  MBAR2(3), MBARM1(130,3), MBARM2(130,3), COV1(3,3),
     +  COV2(3,3), DIF1, P1(3,3), MUT1(130,3), SIM1(3,130),
     +  DD1(3), DD2(3), PA(3,3), PB(3,3), WORK(130), D1, D2,
     +  DIF2, SUM1, SUM2, A1, A2, Z1(3,130), Z2(3,130),
     +  P2(3,3), ZT1(130,3), ZT2(130,3), TBAR, Z1BAR, Z2BAR,
     +  TRKSUM(130), TKSUM2, CT1(3,130), CT2(3,130), SEED,
     +  MUT2(130,3), RESID1(130,3), RESID2(130,3), SQ2(3,3),
```

```
     +  COV1R(3,3), COV2R(3,3), SQ1(3,3), SIMTRK(200,6),
     +  SIM2(3,130), ROTATE(1000,4), DISP(1000,4), DATA(2,4)
C
C...Begin the user input section
C
      WRITE(*,*) 'Enter FNAME.FT of crossover data set'
      WRITE(*.*) '        on disk : '
      READ (*,'(A)') DSNAME
      WRITE(*,*) ' '
      WRITE(*,*) 'What is the NUMBER of the left sensor in'
      WRITE(*.*) ' the crossover pair ? '
      READ(*,*) IDL
      WRITE(*,*) ' '
      WRITE(*,*) 'What is the NUMBER of the right sensor ? '
      READ(*,*) IDR
      WRITE(*,*) ' '
      WRITE(*,*) 'How many simulated tracks do you desire ?'
      WRITE(*.*) '(NOTE : max 1000) '
      READ(*,*) TRACKS
      WRITE(*,*) ' '
      WRITE(*,*) 'Do you want a sample simulated track ?'
      WRITE(*,*) 'Enter 1 for YES, 0 (zero) for NO '
      READ(*,*) TRKOUT
      WRITE(*,*) ' '
      WRITE(*,*) 'Seed for the random number generator'
      WRITE(*,*) 'NOTE : Seed must include a decimal '
      READ(*,*) SEED
      WRITE(*,*) ' '
C
C...Read data from file
C
      OPEN (1,FILE=DSNAME,STATUS='OLD')
      N = 0
 10   N = N + 1
      READ(1,*,END=30, ERR=30) POINT(N),(TRACK(N,I),I=1,6)
C
C...Separate into "left" and "right" sensor tracks
C
      DO 20 I = 1,3
         TR1(N,I) = TRACK(N,I)
         TR2(N,I) = TRACK(N,I+3)
 20   CONTINUE
      GOTO 10
 30   CLOSE (UNIT = 1)
      N = N - 1
C
C...Compute the covariance matrix for each track.
C...-First step, get column averages (with first column
C... average, TBAR, computed for later use)
C
      TBAR = 0.
      DO 50 I = 1,3
```

```
            MBAR1(I) = 0.
            MBAR2(I) = 0.
            DO 40 J = 1,N
               MBAR1(I) = MBAR1(I) + TR1(J,I)
               MBAR2(I) = MBAR2(I) + TR2(J,I)
               IF (I .EQ. 1) TBAR = TBAR + DBLE(POIN1(J))
 40         CONTINUE
            MBAR1(I) = MBAR1(I) / DBLE(N)
            MBAR2(I) = MBAR2(I) / DBLE(N)
 50      CONTINUE
         TBAR = TBAR / DBLE(N)
C
C...Do the matrix multiplication : A(t)xA , subtracting off
C...the mean from each column entry to form the covariance
C...matrix.  Also make the matrix of (points - means) for
C...later use.
C
         DO 80 I = 1,3
          DO 70 J = 1,3
           COV1(I,J) = 0.
           COV2(I,J) = 0.
           DO 60 K = 1,N
            COV1(I,J) = COV1(I,J)+(TR1(K,I)-MBAR1(I))
     *                           *(TR1(K,J)-MBAR1(J))
            COV2(I,J) = COV2(I,J)+(TR2(K,I)-MBAR2(I))
     *                           *(TR2(K,J)-MBAR2(J))
 60         CONTINUE
            COV1(I,J) = COV1(I,J) / DBLE(N-1)
            COV2(I,J) = COV2(I,J) / DBLE(N-1)
 70        CONTINUE
 80      CONTINUE
         DO 100 I = 1,N
            DO 90 J = 1,3
               MBARM1(I,J) = TR1(I,J) - MBAR1(J)
               MBARM2(I,J) = TR2(I,J) - MBAR2(J)
 90         CONTINUE
 100     CONTINUE
C
C...Form the matrix P of ordered principle components for
C...each track.  The columns of P are the eigenvectors
C...associated with the eigen-values of the covariance
C...matrix for each track arranged in order of descending
C...eigenvalues.  (ie. the eigenvector associated with
C...the largest eigenvalue is the first column)
C
C...Call routine to compute eigenvalues/vectors for each
C...track
C
         CALL EIGRS(COV1,3,11,DD1,PA,3,WORK,IER)
         CALL EIGRS(COV2,3,11,DD2,PB,3,WORK,IER)
C
C...Get eigenvectors in eigenvalue order, largest to
```

```
C...smallest, by column
C
C...DD1/2 = eigenvalue vector
C...PA/PB = eigenvector matrices
C
      BIG1(1) = 1
      BIG2(1) = 1
      BIG1(3) = 3
      BIG2(3) = 3
      IF (DD1(BIG1(1)) .LT. DD1(2)) BIG1(1) = 2
      IF (DD2(BIG2(1)) .LT. DD2(2)) BIG2(1) = 2
      IF (DD1(BIG1(1)) .LT. DD1(3)) BIG1(1) = 3
      IF (DD2(BIG2(1)) .LT. DD2(3)) BIG2(1) = 3
      IF (DD1(BIG1(3)) .GT. DD1(1)) BIG1(3) = 1
      IF (DD2(BIG2(3)) .GT. DD2(1)) BIG2(3) = 1
      IF (DD1(BIG1(3)) .GT. DD1(2)) BIG1(3) = 2
      IF (DD2(BIG2(3)) .GT. DD2(2)) BIG2(3) = 2
      IF ((BIG1(1) + BIG1(3)) .EQ. 3) THEN
         BIG1(2) = 3
      ELSE
         IF ((BIG1(1) + BIG1(3)) .EQ. 4) THEN
            BIG1(2) = 2
         ELSE
            BIG1(2) = 1
         END IF
      END IF
      IF ((BIG2(1) + BIG2(3)) .EQ. 3) THEN
         BIG2(2) = 3
      ELSE
         IF ((BIG2(1) + BIG2(3)) .EQ. 4) THEN
            BIG2(2) = 2
         ELSE
            BIG2(2) = 1
         END IF
      END IF
      DO 130 I = 1,3
         DO 120 J = 1,3
            P1(I,J) = PA(I,BIG1(J))
            P2(I,J) = PB(I,BIG2(J))
 120     CONTINUE
 130  CONTINUE
C
C...Compute the matrix ZT for each track
C...ZT represents the projection of the track data onto
C...the principle components
C...ZT = (TR - MBAR) x P = MBARM x P
C...where TR - MBAR = track data minus the column average
C...for each row
C
C...Call routine to multiply matrices AxB
C
      CALL VMULFF(MBARM1,P1,N,3,3,130,3,ZT1,130,IER)
```

```
      CALL VMULFF(MBARM2,P2,N,3,3,130,3,ZT2,130,IER)
C
C...Since there are some points missing from the data set,
C...perform a simple least squares linear regression onto
C...the first principle component
C
      TKSUM2 = 0.0
      Z1BAR = 0.0
      Z2BAR = 0.0
      DO 140 I = 1,N
         Z1BAR = Z1BAR + ZT1(I,1)
         Z2BAR = Z2BAR + ZT2(I,1)
         TRKSUM(I) = DBLE(POINT(I)) - TBAR
         TKSUM2 = TKSUM2 + TRKSUM(I)**2
 140  CONTINUE
      Z1BAR = Z1BAR / DBLE(N)
      Z2BAR = Z2BAR / DBLE(N)
      SUM1 = 0.0
      SUM2 = 0.0
      DO 150 I = 1,N
         DIF1 = (ZT1(I,1) - Z1BAR) * TRKSUM(I)
         DIF2 = (ZT2(I,1) - Z2BAR) * TRKSUM(I)
         SUM1 = SUM1 + DIF1
         SUM2 = SUM2 + DIF2
 150  CONTINUE
      D1 = SUM1 / TKSUM2
      D2 = SUM2 / TKSUM2
      A1 = Z1BAR - D1 * TBAR
      A2 = Z2BAR - D2 * TBAR
      DO 160 I = 1,N
         ZT1(I,1) = A1 + D1 * DBLE(POINT(I))
         ZT2(I,1) = A2 + D2 * DBLE(POINT(I))
 160  CONTINUE
C
C...Get CT matrix which represents the orthogonal projection
C...of the data onto the straight line of the first
C...principle component
C
      DO 170 I = 1,N
         Z1(1,I) = ZT1(I,1)
         Z2(1,I) = ZT2(I,1)
 170  CONTINUE
      DO 180 I = 1,N
         Z1(2,I) = 0.0
         Z1(3,I) = 0.0
         Z2(2,I) = 0.0
         Z2(3,I) = 0.0
 180  CONTINUE
C
C...Call routine to multiply matrices AxB
C
      CALL VMULFF(P1,Z1,3,3,N,3,3,CT1,3,IER)
```

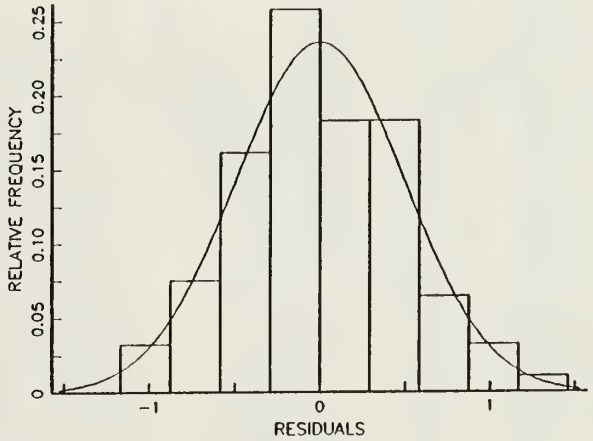
```
      CALL VMULFF(P2,Z2,3,3,N,3,3,CT2,3,IER)
C
C...Move "line" of data back into original coordinate system
C
      DO 200 I = 1,N
         DO 190 J = 1,3
            MUT1(I,J) = CT1(J,I) + MBAR1(J)
            MUT2(I,J) = CT2(J,I) + MBAR2(J)
C
C...Compute residuals for each track
C
            RESID1(I,J) = TR1(I,J) - MUT1(I,J)
            RESID2(I,J) = TR2(I,J) - MUT2(I,J)
 190     CONTINUE
 200  CONTINUE
C
C...Write the set of residuals out to the file RESIDUAL.DAT
C
      OPEN (2, FILE = 'RESIDUAL.DAT', STATUS = 'NEW')
      DO 210 I = 1,N
        WRITE(2,330) (RESID1(I,J),J=1,3),(RESID2(I,J),J=1,3)
 210  CONTINUE
C
C...Compute the covariance matrix of the residuals
C...(Note : column averages are identically zero)
C
C...Call routine to multiply matrices A(t)xB, then divide
C...by N-1
C
      CALL VMULFM(RESID1,RESID1,N,3,3,130,130,COV1R,3,IER)
      CALL VMULFM(RESID2,RESID2,N,3,3,130,130,COV2R,3,IER)
      DO 230 I = 1,3
         DO 220 J = 1,3
            COV1R(I,J) = COV1R(I,J) / DBLE(N-1)
            COV2R(I,J) = COV2R(I,J) / DBLE(N-1)
 220     CONTINUE
 230  CONTINUE
C
C...Get "square root" of covariance matrix of residuals
C
      SQ1(3,3) = DSQRT(COV1R(3,3))
      SQ2(3,3) = DSQRT(COV2R(3,3))
      SQ1(2,3) = COV1R(2,3) / SQ1(3,3)
      SQ2(2,3) = COV2R(2,3) / SQ2(3,3)
      SQ1(2,2) = DSQRT(COV1R(2,2) - SQ1(2,3)**2)
      SQ2(2,2) = DSQRT(COV2R(2,2) - SQ2(2,3)**2)
      SQ1(1,3) = COV1R(1,3) / SQ1(3,3)
      SQ2(1,3) = COV2R(1,3) / SQ2(3,3)
      SQ1(1,2) = (COV1R(1,2) - SQ1(1,3)*SQ1(2,3)) / SQ1(2,2)
      SQ2(1,2) = (COV2R(1,2) - SQ2(1,3)*SQ2(2,3)) / SQ2(2,2)
      SQ1(1,1) = DSQRT(COV1R(1,1)-(SQ1(1,2)**2+SQ1(1,3)**2))
      SQ2(1,1) = DSQRT(COV2R(1,1)-(SQ2(1,2)**2+SQ2(1,3)**2))
```

```
      SQ1(2,1) = 0.
      SQ2(2,1) = 0.
      SQ1(3,2) = 0.
      SQ2(3,2) = 0.
      SQ1(3,1) = 0.
      SQ2(3,1) = 0.
C
C...Compute sets of residuals and get rotation/displacement
C...values
C
      DO 410 SIMS = 1,TRACKS
C
C...Compute set of simulated residuals from normal(0,1)
C...deviates
C
      DO 260 I = 1,3
C
C...Call routine to generate Normal(0,1) deviates
C
         CALL GGNPM(SEED,2*N,NORM)
         DO 250 J = 1,N
            RESID1(J,I) = NORM(J)
            RESID2(J,I) = NORM(N+J)
 250     CONTINUE
 260  CONTINUE
C
C...Call routine to multiply matrices AxB(t)
C
      CALL VMULFP(SQ1,RESID1,3,3,N,3,130,SIM1,3,IER)
      CALL VMULFP(SQ2,RESID2,3,3,N,3,130,SIM2,3,IER)
C
C...Put together Nx6 matrix of simulated tracks for both
C...arrays by adding straight line in original coordinate
C...system to residuals
C
      DO 280 I = 1,N
         DO 270 J = 1,3
            SIMTRK(I,J) = SIM1(J,I) + MUT1(I,J)
            SIMTRK(I,J+3) = SIM2(J,I) + MUT2(I,J)
 270     CONTINUE
 280  CONTINUE
C
C...Write the first simulated track out to the file
C...SIMTRACK.DAT if a sample simulated track was requested.
C
      IF((SIMS .EQ. 1) .AND. (TRKOUT .GT. 0)) THEN
         OPEN (3, FILE = 'SIMTRACK.DAT', STATUS = 'NEW')
         DO 285 I = 1,N
            WRITE(3,340) (SIMTRK(I,J),J = 1,6)
 285     CONTINUE
      END IF
C
```

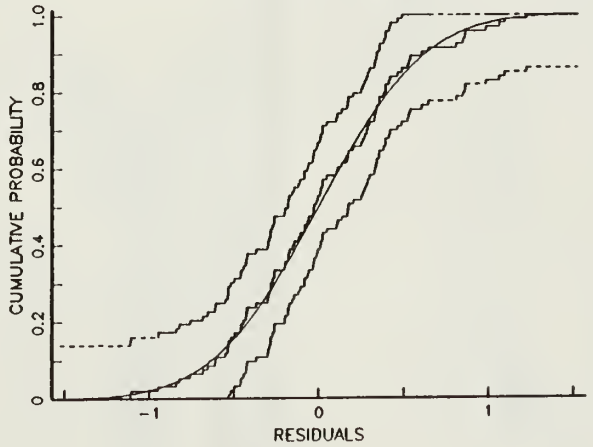
```
C...Feed the simulated track into KEYMAIN to get rotation
C...and displacement numbers
C
      CALL KEYSUB(SIMTRK,N,DATA,IDL,IDR)
C
C...Make 2 matrices - one for displacement data and one for
C...the rotation data
C
      DO 290 I = 1,4
         DISP(SIMS,I) = DATA(1,I)
         ROTATE(SIMS,I) = DATA(2,I)
 290  CONTINUE
      WRITE(*,320) SIMS
C
C...Go back and do it again
C
 410  CONTINUE
C
C...After TRACKS simulated tracks, write the displacement
C...sets and the rotation sets out to a file
C
      OPEN(4,FILE = 'DISPLACE.DAT',STATUS = 'NEW')
      OPEN(5,FILE = 'ROTATE.DAT',STATUS = 'NEW')
      DO 300 I = 1,TRACKS
         WRITE(4,310) (DISP(I,J),J = 1,4)
         WRITE(5,310) (ROTATE(I,J),J = 1,4)
 300  CONTINUE
C
C...Close out the files
C
      CLOSE(UNIT = 2)
      CLOSE(UNIT = 3)
      CLOSE(UNIT = 4)
      CLOSE(UNIT = 5)
      STOP
 310  FORMAT(2X,4F17.8)
 320  FORMAT(2X,'Through KEYMAIN ',I6,' time(s) so far')
 330  FORMAT(1X,6F12.7)
 340  FORMAT(1X,6F12.5)
      END
```

DATA SET D4 / SENSOR A / X RESIDUALS

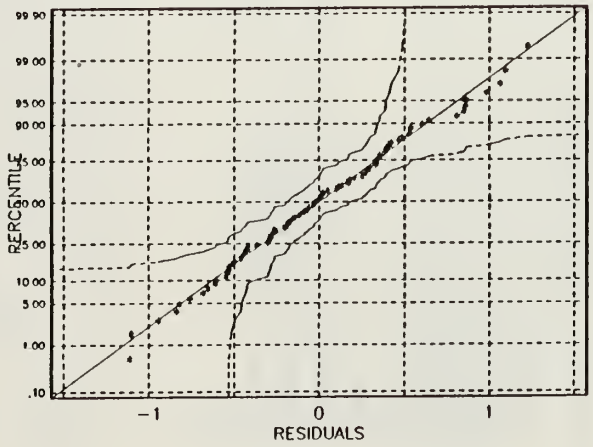
NORMAL DENSITY FUNCTION, N=93



NORMAL CUMULATIVE DISTRIBUTION FUNCTION, N=93



NORMAL PROBABILITY PLOT



NORMAL DISTRIBUTION

X : D4ZR[:1]
SELECTION : ALL
LABEL : RESIDUALS
SAMPLE SIZE: 93
MINIMUM : -1.113
MAXIMUM : 1.228
CENSORING : NONE
EST. METHOD: MAXIMUM LIKELIHOOD

	SAMPLE	FITTED
MEAN :	1.8205E-11	1.8205E-11
STD DEV :	4.9458E-1	4.9456E-1
SKEWNESS:	1.5482E-1	0.0000E0
KURTOSIS:	2.7772E0	3.0000E0

COVARIANCE MATRIX OF PARAMETER ESTIMATES

	MU	SIGMA
MU	0.0026017	0
SIGMA	0	0.001315

PERCENTILES	SAMPLE	FITTED
5:	-0.82182	-8.1365E-1
10:	-0.80705	-8.3389E-1
25:	-0.30219	-3.3342E-1
50:	-0.028901	4.9970E-8
75:	0.34041	3.3342E-1
90:	0.80752	8.3389E-1
95:	0.88492	8.1365E-1

GOODNESS OF FIT

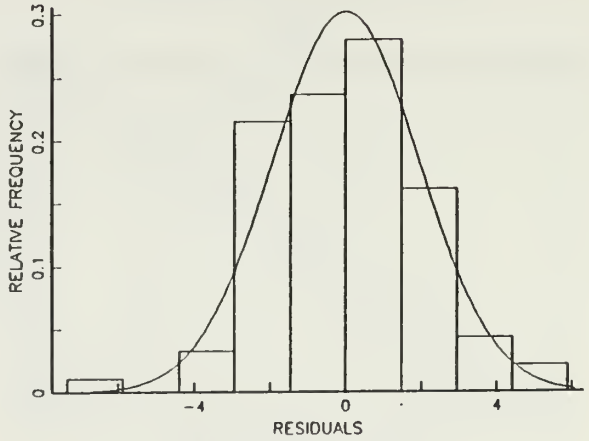
CHI-SQUARE :	1.9993
DEG FREED:	5
SIGNIF :	0.84924
KOLM-SMIRN :	0.051013
SIGNIF :	0.96888
CRAMER-V M :	0.00002551
SIGNIF :	> .15
ANDER-DARL :	0.21289
SIGNIF :	> .15

KS, AD, AND CV SIGNIF. LEVELS NOT EXACT WITH ESTIMATED PARAMETERS

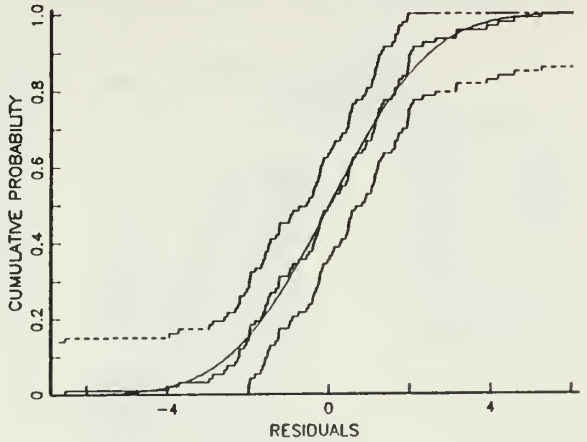
		0.95 CONFIDENCE INTERVALS	
PARAMETER	ESTIMATE	LOWER	UPPER
MU	1.8205E-11	-0.10147	0.10147
SIGMA	4.9458E-1	0.43228	0.57802

DATA SET D4 / SENSOR A / Y RESIDUALS

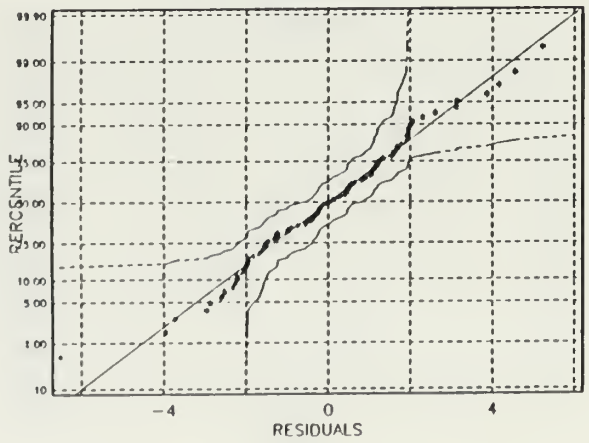
NORMAL DENSITY FUNCTION, N=93



NORMAL CUMULATIVE DISTRIBUTION FUNCTION, N=93



NORMAL PROBABILITY PLOT

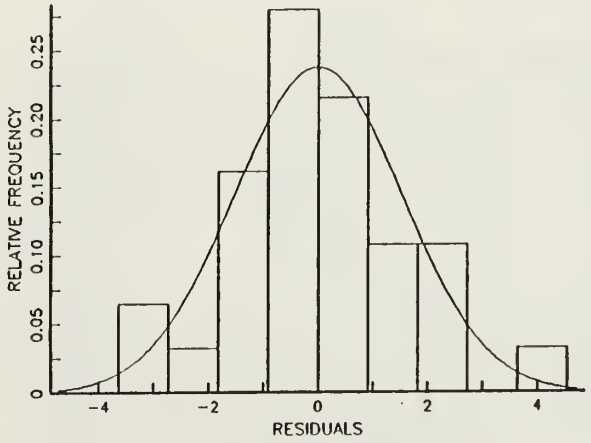


NORMAL DISTRIBUTION

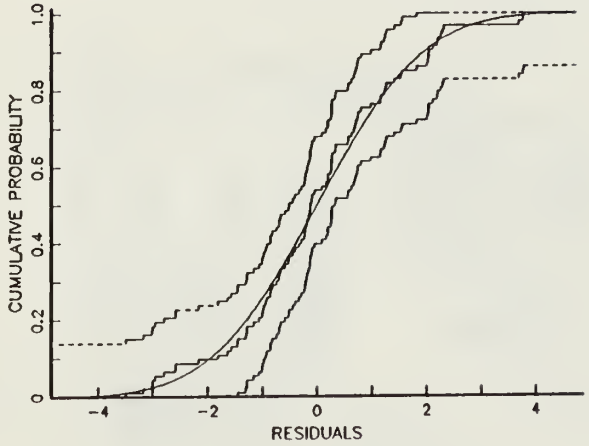
```
X           : 042R[:2]
SELECTION   : ALL
LABEL       : RESIDUALS
SAMPLE SIZE: 93
MINIMUM     :  -8.529
MAXIMUM     :   5.228
CENSORING   : NONE
EST. METHOD: MAXIMUM LIKELIHOOD

          SAMPLE       FITTED
MEAN    : -1.2605E-11  -1.2605E-11
STD DEV :  1.9446E0     1.9446E0
SKEWNESS: -8.2178E-2    0.0000E0
KURTOSIS:  3.6756E0     3.0000E0

PERCENTILES SAMPLE     FITTED
    5:      -2.8624    -3.1993E0
   10:      -2.2194    -2.4925E0
   25:      -1.5088    -1.3110E0
   50:       0.014385   1.9641E-7
   75:       1.2087     1.3110E0
   90:       1.9979     2.4925E0
   95:       3.1388     3.1993E0

COVARIANCE MATRIX OF
PARAMETER ESTIMATES
             MU          SIGMA
MU       0.040224  0
SIGMA    0         0.020331

GOODNESS OF FIT
CHI-SQUARE :   3.015
DEG FREED:     3
SIGNIF     :   0.38932
KOLM-SMIRN :   0.058418
SIGNIF     :   0.80878
CRAMER-V M :   0.0000356819
SIGNIF     :   > .15
ANDER-DARL :   0.33658
SIGNIF     :   > .15
KS, AD, AND CV SIGNIF. LEVELS NOT EXACT WITH ESTIMATED PARAMETERS

                    0.95 CONFIDENCE INTERVALS
PARAMETER  ESTIMATE      LOWER      UPPER
MU         -1.2605E-11   -0.39899    0.39899
SIGMA       1.9446E0      1.6998     2.2728
```

DATA SET D4 / SENSOR A / Z RESIDUALS

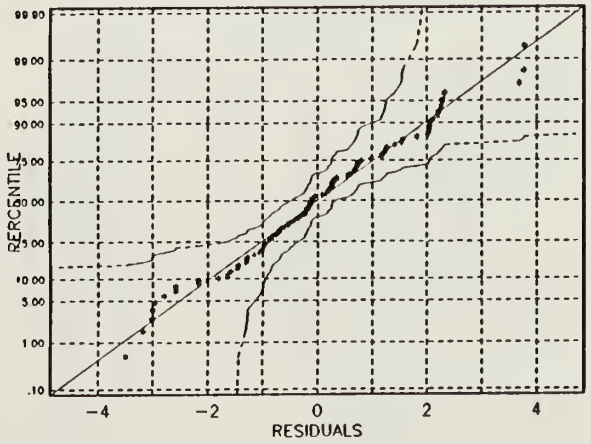
NORMAL DENSITY FUNCTION, N=93



NORMAL CUMULATIVE DISTRIBUTION FUNCTION, N=93



NORMAL PROBABILITY PLOT



NORMAL DISTRIBUTION

X : D4ZR[:3]
SELECTION : ALL
LABEL : RESIDUALS
SAMPLE SIZE: 93
MINIMUM : -3.497
MAXIMUM : 3.777
CENSORING : NONE
EST. METHOD: MAXIMUM LIKELIHOOD

	SAMPLE	FITTED
MEAN :	-5.0242E-13	-5.0242E-13
STD DEV :	1.5289E0	1.5289E0
SKEWNESS:	8.2124E-2	0.0000E0
KURTOSIS:	3.0747E0	3.0000E0

PERCENTILES	SAMPLE	FITTED
5:	-2.9583	-2.5154E0
10:	-1.8089	-1.9596E0
25:	-0.91771	-1.0308E0
50:	-0.11995	1.5443E-7
75:	0.80299	1.0308E0
90:	2.0501	1.9596E0
95:	2.2884	2.5154E0

COVARIANCE MATRIX OF PARAMETER ESTIMATES

	MU	SIGMA
MU	0.024865	0
SIGMA	0	0.012587

GOODNESS OF FIT

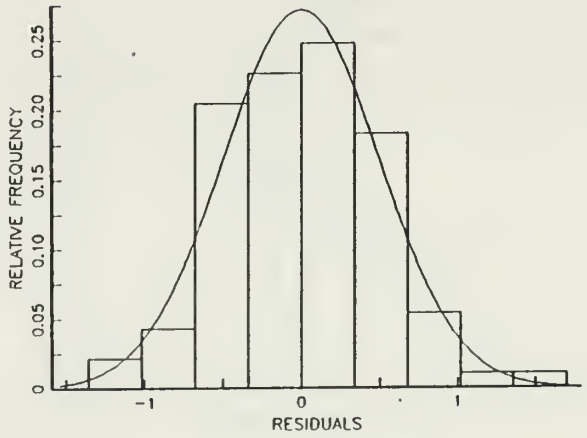
CHI-SQUARE :	8.3189
DEG FREED:	5
SIGNIF :	0.13951
KOLM-SMIRN :	0.066238
SIGNIF :	0.80917
CRAMER-V M :	0.0000578O1
SIGNIF :	> .15
ANDER-DARL :	0.48755
SIGNIF :	> .15

KS, AD, AND CV SIGNIF. LEVELS NOT EXACT WITH ESTIMATED PARAMETERS

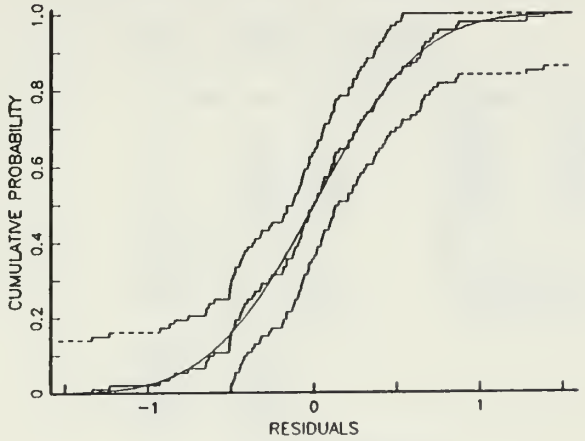
		0.95 CONFIDENCE INTERVALS	
PARAMETER	ESTIMATE	LOWER	UPPER
MU	-5.0242E-13	-0.31389	0.31389
SIGMA	1.5289E0	1.3363	1.7889

DATA SET D4 / SENSOR B / X RESIDUALS

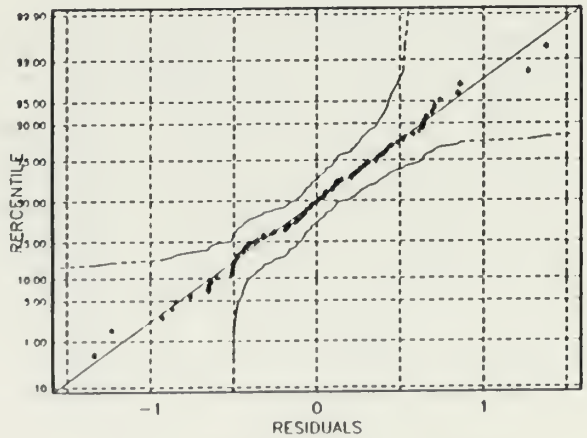
NORMAL DENSITY FUNCTION, N=93



NORMAL CUMULATIVE DISTRIBUTION FUNCTION, N=93



NORMAL PROBABILITY PLOT



NORMAL DISTRIBUTION

X : D4ZR[:4]
SELECTION : ALL
LABEL : RESIDUALS
SAMPLE SIZE: 93
MINIMUM : -1.338
MAXIMUM : 1.377
CENSORING : NONE
EST. METHOD: MAXIMUM LIKELIHOOD

	SAMPLE	FITTED
MEAN :	2.5231E-12	2.5231E-12
STD DEV :	4.9919E-1	4.9919E-1
SKEWNESS:	1.3670E-2	0.0000E0
KURTOSIS:	3.2045E0	3.0000E0

COVARIANCE MATRIX OF PARAMETER ESTIMATES

	MU	SIGMA
MU	0.0026508	0
SIGMA	0	0.0013397

PERCENTILES	SAMPLE	FITTED
5:	-0.84395	-8.2127E-1
10:	-0.80177	-6.3983E-1
25:	-0.37422	-3.3855E-1
50:	0.0015618	5.0424E-6
75:	0.35292	3.3655E-1
90:	0.64081	8.3983E-1
95:	0.73788	8.2127E-1

GOODNESS OF FIT

CHI-SQUARE :	2.2871
DEG FREED:	3
SIGNIF :	0.51888
KOLM-SMIRN :	0.044111
SIGNIF :	0.99355
CRAMER-V M :	0.000022793
SIGNIF :	> .15
ANDER-DARL :	0.19988
SIGNIF :	> .15

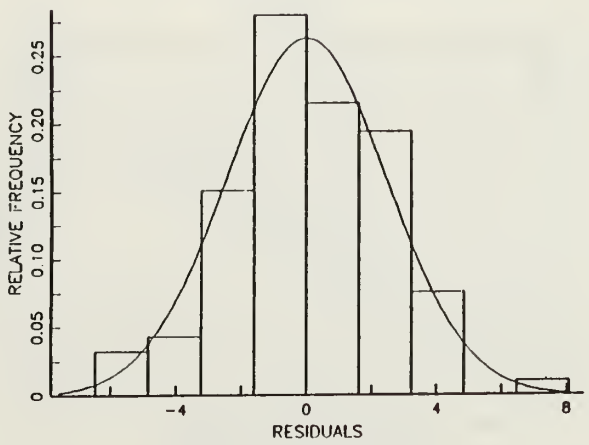
KS, AD, AND CV SIGNIF. LEVELS NOT EXACT WITH ESTIMATED PARAMETERS

0.95 CONFIDENCE INTERVALS

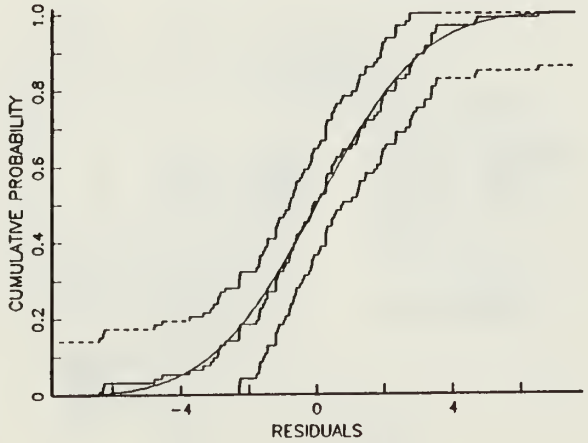
PARAMETER	ESTIMATE	LOWER	UPPER
MU	2.5231E-12	-0.10242	0.10242
SIGMA	4.9919E-1	0.43831	0.58343

DATA SET D4 / SENSOR B / Y RESIDUALS

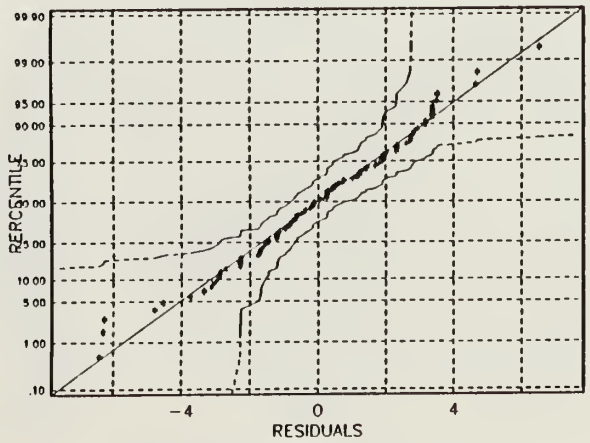
NORMAL DENSITY FUNCTION, N=93



NORMAL CUMULATIVE DISTRIBUTION FUNCTION, N=93



NORMAL PROBABILITY PLOT



NORMAL DISTRIBUTION

X	: D42R[;3]
SELECTION	: ALL
LABEL	: RESIDUALS
SAMPLE SIZE:	93
MINIMUM	: -8.398
MAXIMUM	: 6.519
CENSORING	: NONE
EST. METHOD:	MAXIMUM LIKELIHOOD

	SAMPLE	FITTED
MEAN	: -1.3601E-11	-1.3601E-11
STD DEV	: 2.4539E0	2.4539E0
SKEWNESS:	-2.3291E-1	0.0000E0
KURTOSIS:	3.2253E0	3.0000E0

PERCENTILES	SAMPLE	FITTED
5:	-4.5443	-4.0373E0
10:	-2.9149	-3.1453E0
25:	-1.4521	-1.8544E0
50:	-0.095774	2.4785E-7
75:	1.8869	1.8544E0
90:	3.1538	3.1453E0
95:	3.4892	4.0373E0

COVARIANCE MATRIX OF PARAMETER ESTIMATES

	MU	SIGMA
MU	0.064055	0
SIGMA	0	0.032378

GOODNESS OF FIT

CHI-SQUARE :	1.8814
DEG FREED:	3
SIGNIF :	0.59738
KOLM-SMIRN :	0.051889
SIGNIF :	0.98369
CRAMER-V M :	0.000027418
SIGNIF :	> .15
ANDER-DARL :	0.29031
SIGNIF :	. > .15

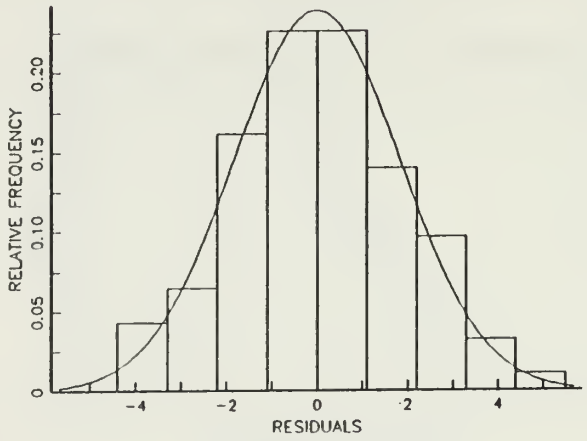
KS, AD, AND CV SIGNIF. LEVELS NOT EXACT WITH ESTIMATED PARAMETERS

0.95 CONFIDENCE INTERVALS

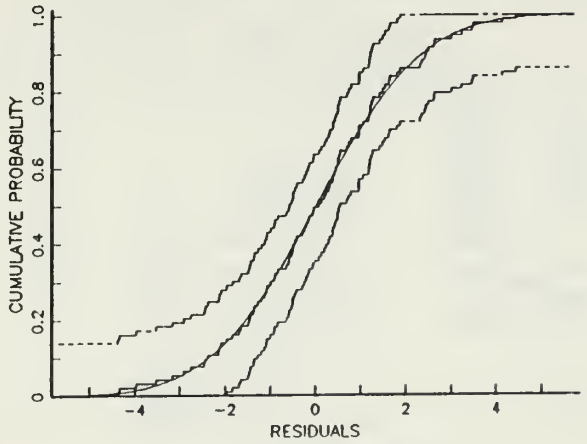
PARAMETER	ESTIMATE	LOWER	UPPER
MU	-1.3601E-11	-0.50349	0.50349
SIGMA	2.4539E0	2.1448	2.8681

DATA SET D4 / SENSOR B / Z RESIDUALS

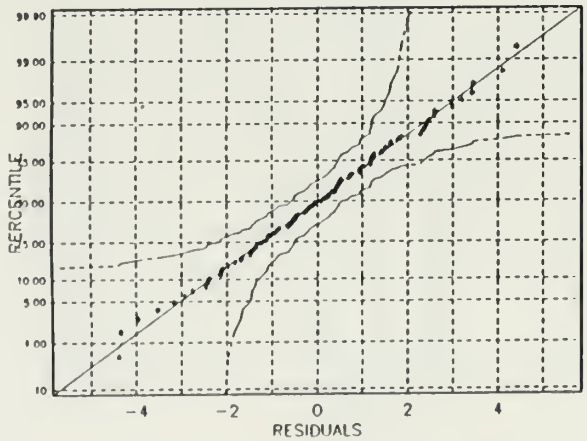
NORMAL DENSITY FUNCTION, N=93



NORMAL CUMULATIVE DISTRIBUTION FUNCTION, N=93



NORMAL PROBABILITY PLOT

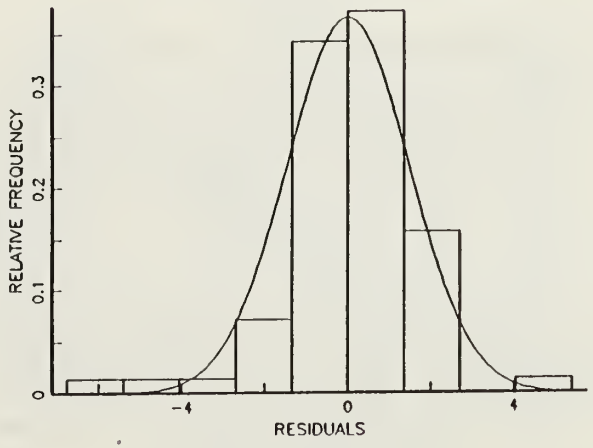


NORMAL DISTRIBUTION

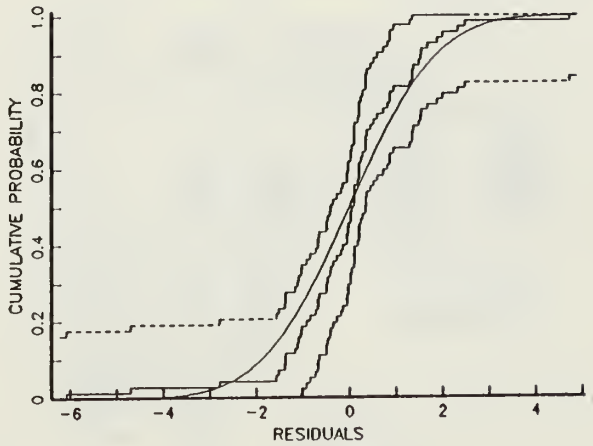
```
X           :  D42R[;8]
SELECTION   1  ALL
LABEL       :  RESIDUALS
SAMPLE SIZE:  93
MINIMUM     :   -4.381
MAXIMUM     :   4.408
CENSORING   :  NONE
EST. METHOD:  MAXIMUM LIKELIHOOD

          SAMPLE          FITTED
MEAN    : -4 9937E-13   -4 9937E-13
STD DEV :  1 8381E0      1 8381E0
SKEWNESS: -4 6678E-2     0 0000E0
KURTOSIS:  2 8645E0      3.0000E0

PERCENTILES SAMPLE      FITTED
    5:      -3 1597     -3 0240E0
   10:      -2 3855     -2 3559E0
   25:      -1 205      -1 2392E0
   50:       0 07021     1 8566E-7
   75:       1.1877      1 2392E0
   90:       2.4176      2 3559E0
   95:       3.1991      3 0240E0

KS, AD, AND CV SIGNIF  LEVELS NOT EXACT WITH ESTIMATED PARAMETERS

                    0.95 CONFIDENCE INTERVALS
PARAMETER  ESTIMATE      LOWER      UPPER
MU         -4 9937E-13  -0 37713   0 37713
SIGMA       1 8381E0     1 8065    2 1483

COVARIANCE MATRIX OF
PARAMETER ESTIMATES
            MU          SIGMA
MU      0 035938 0
SIGMA 0             0 018184

GOODNESS OF FIT
CHI-SQUARE :      1.051
  DEG FREED:      5
  SIGNIF   :      0 95835
KOLM-SMIRN :      0 032635
  SIGNIF   :      0 99997
CRAMER-V M :      0 000010893
  SIGNIF   :      > .15
ANDER-DARL :      0 10317
  SIGNIF   :      > .15
```

DATA SET D2A2 / SENSOR A / X RESIDUALS

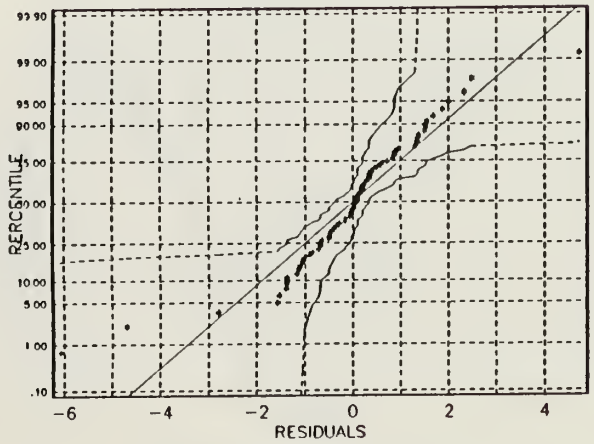
NORMAL DENSITY FUNCTION, N=70



NORMAL CUMULATIVE DISTRIBUTION FUNCTION, N=70



NORMAL PROBABILITY PLOT



NORMAL DISTRIBUTION

X : 02A2ZR[;1]
SELECTION : ALL
LABEL : RESIDUALS
SAMPLE SIZE: 70
MINIMUM : -6.075
MAXIMUM : 4.725
CENSORING : NONE
EST. METHOD: MAXIMUM LIKELIHOOD

	SAMPLE	FITTED
MEAN :	7.5358E-13	7.5358E-13
STD DEV :	1.4729E0	1.4729E0
SKEWNESS:	-8.8663E-1	0.0000E0
KURTOSIS:	7.8266E0	3.0000E0

PERCENTILES	SAMPLE	FITTED
5:	-1.5715	-2.4232E0
10:	-1.3752	-1.8878E0
25:	-0.87127	-9.9299E-1
50:	0.053668	1.4877E-7
75:	0.72741	9.9299E-1
90:	1.5253	1.8878E0
95:	2.0014	2.4232E0

COVARIANCE MATRIX OF PARAMETER ESTIMATES

	MU	SIGMA
MU	0.030548	0
SIGMA	0	0.015495

GOODNESS OF FIT

CHI-SQUARE :	2.3735
DEG FREEDOM:	1
SIGNIF :	0.12341
KOLM-SMIRN :	0.10013
SIGNIF :	0.4841
CRAMER-V M :	0.00030592
SIGNIF :	> .15
ANDER-DARL :	1.6863
SIGNIF :	< .15

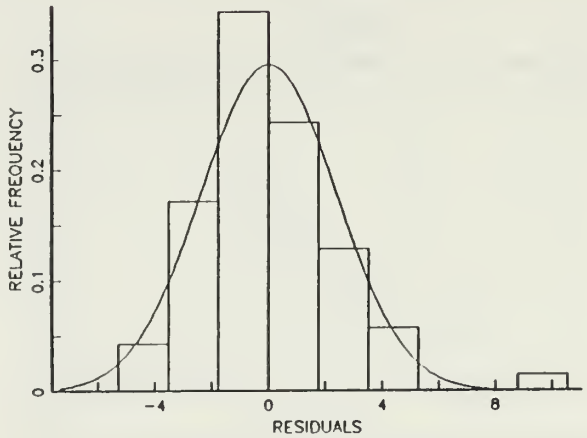
KS, AD, AND CV SIGNIF. LEVELS NOT EXACT WITH ESTIMATED PARAMETERS

0.95 CONFIDENCE INTERVALS

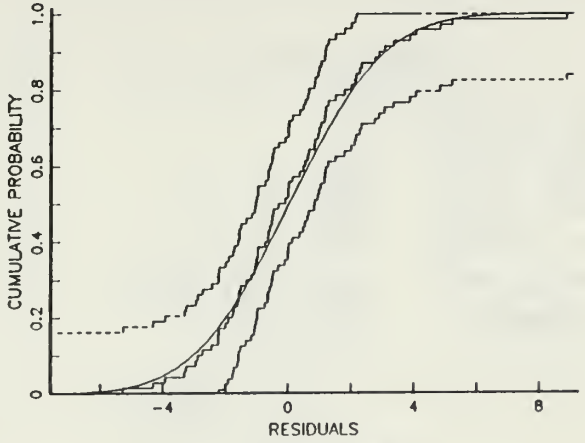
PARAMETER	ESTIMATE	LOWER	UPPER
MU	7.5358E-13	-0.34883	0.34883
SIGMA	1.4729E0	1.2628	1.7674

DATA SET D2A2 / A SENSOR / Y RESIDUALS

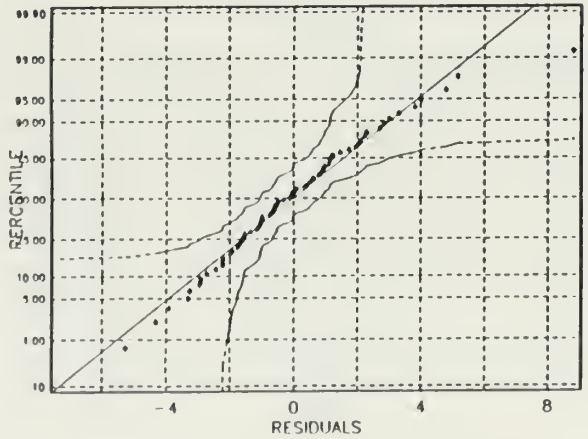
NORMAL DENSITY FUNCTION, N=70



NORMAL CUMULATIVE DISTRIBUTION FUNCTION, N=70



NORMAL PROBABILITY PLOT



NORMAL DISTRIBUTION

X : D2A22R[:2]
SELECTION : ALL
LABEL : RESIDUALS
SAMPLE SIZE: 70
MINIMUM : -5.271
MAXIMUM : 8.804
CENSORING : NONE
EST. METHOD: MAXIMUM LIKELIHOOD

	SAMPLE	FITTED
MEAN :	0.4233E-13	0.4233E-13
STD DEV :	2.3783E0	2.3783E0
SKEWNESS:	7.3968E-1	0.0000E0
KURTOSIS:	4.5347E0	3.0000E0

PERCENTILES	SAMPLE	FITTED
5:	-3.3138	-3.9129E0
10:	-2.8968	-3.0484E0
25:	-1.5818	-1.6034E0
50:	-0.33134	2.4023E-7
75:	1.1584	1.6034E0
90:	2.8499	3.0484E0
95:	4.0107	3.9129E0

COVARIANCE MATRIX OF PARAMETER ESTIMATES

	MU	SIGMA
MU	0.079852	0
SIGMA	0	0.040403

GOODNESS OF FIT

CHI-SQUARE :	2.7709
DEG FREED:	3
SIGNIF :	0.42833
KOLM-SMIRN :	0.074584
SIGNIF :	0.83098
CRAMER-V M :	0.0000720980
SIGNIF :	> .15
ANDER-DARL :	0.39788
SIGNIF :	> .15

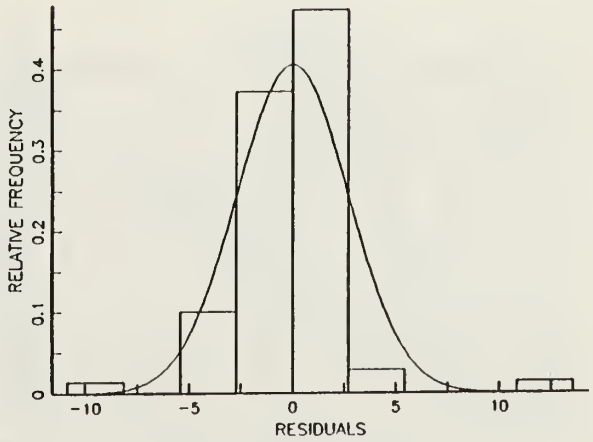
KS, AD, AND CV SIGNIF. LEVELS NOT EXACT WITH ESTIMATED PARAMETERS

0.95 CONFIDENCE INTERVALS

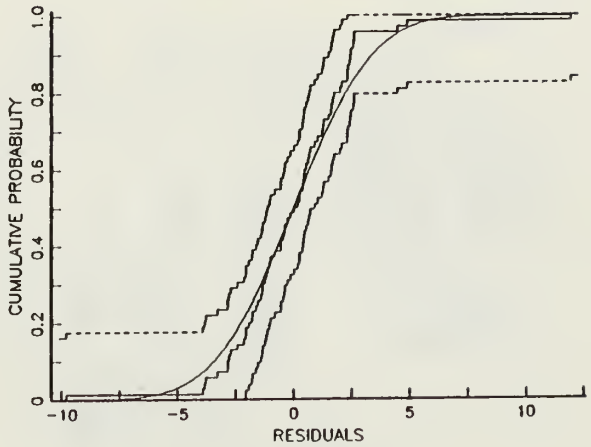
PARAMETER	ESTIMATE	LOWER	UPPER
MU	0.4233E-13	-0.56328	0.56328
SIGMA	2.3783E0	2.0392	2.854

DATA SET D2A2 / A SENSOR / Z RESIDUALS

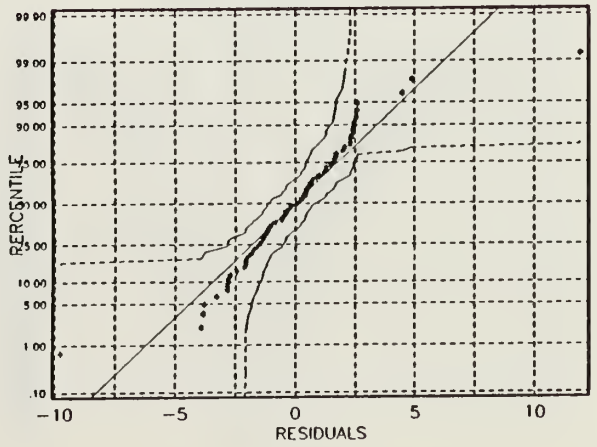
NORMAL DENSITY FUNCTION, N=70



NORMAL CUMULATIVE DISTRIBUTION FUNCTION, N=70



NORMAL PROBABILITY PLOT



NORMAL DISTRIBUTION

X : 02A2ZR[:3]
SELECTION : ALL
LABEL : RESIDUALS
SAMPLE SIZE: 70
MINIMUM : -8.780
MAXIMUM : 11.929
CENSORING : NONE
EST. METHOD: MAXIMUM LIKELIHOOD

	SAMPLE	FITTED
MEAN :	-2.5986E-14	-2.5986E-14
STD DEV :	2.6746E0	2.6746E0
SKEWNESS:	5.7729E-1	0.0000E0
KURTOSIS:	8.9184E0	3.0000E0

PERCENTILES	SAMPLE	FITTED
5:	-3.7719	-4.4003E0
10:	-2.828	-3.4281E0
25:	-1.5327	-1.8032E0
50:	0.032141	2.7015E-7
75:	1.6011	1.8032E0
90:	2.4491	3.4281E0
95:	2.8071	4.4003E0

COVARIANCE MATRIX OF PARAMETER ESTIMATES

	MU	SIGMA
MU	0.10073	0
SIGMA	0	0.051098

GOODNESS OF FIT

CHI-SQUARE :	9.8385
DEG FREED:	1
SIGNIF :	0.001709
KOLM-SMIRN :	0.12198
SIGNIF :	0.24858
CRAMER-V M :	0.0001348
SIGNIF :	> .15
ANDER-DARL :	1.0498
SIGNIF :	> .15

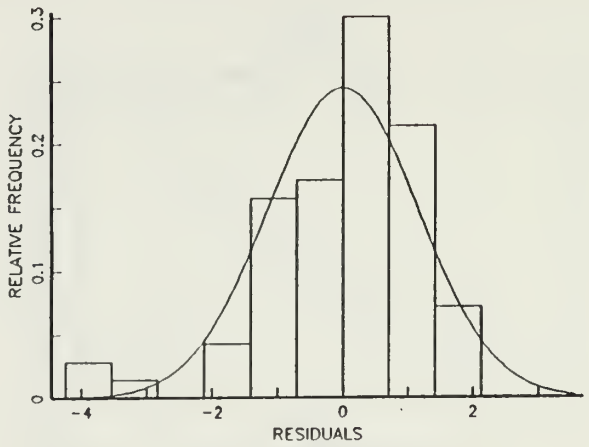
KS, AD, AND CV SIGNIF. LEVELS NOT EXACT WITH ESTIMATED PARAMETERS

0.95 CONFIDENCE INTERVALS

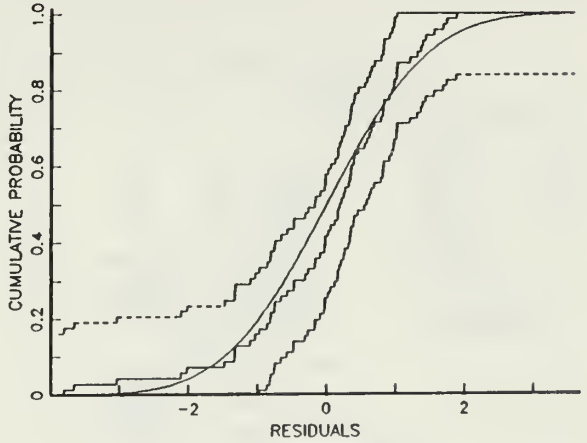
PARAMETER	ESTIMATE	LOWER	UPPER
MU	-2.5986E-14	-0.63344	0.63344
SIGMA	2.6746E0	2.2932	3.2095

DATA SET D2A2 / B SENSOR / X RESIDUALS

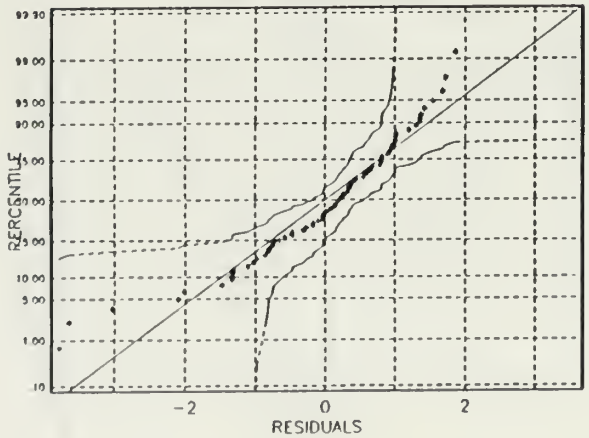
NORMAL DENSITY FUNCTION, N=70



NORMAL CUMULATIVE DISTRIBUTION FUNCTION, N=70



NORMAL PROBABILITY PLOT



NORMAL DISTRIBUTION

X	: D2A2ZR[:4]
SELECTION	: ALL
LABEL	: RESIDUALS
SAMPLE SIZE:	70
MINIMUM	: -3.793
MAXIMUM	: 1.862
CENSORING	: NONE
EST. METHOD:	MAXIMUM LIKELIHOOD

	SAMPLE	FITTED
MEAN	: 1.0854E-12	1.0854E-12
STD DEV	: 1.1548E0	1.1546E0
SKEWNESS:	-1.1488E0	0.0000E0
KURTOSIS:	4.6943E0	3.0000E0

COVARIANCE MATRIX OF PARAMETER ESTIMATES

	MU	SIGMA
MU	0.018773	0
SIGMA	0	0.0095224

PERCENTILES	SAMPLE	FITTED
5:	-2.0980	-1.8998E0
10:	-1.331	-1.4793E0
25:	-0.85788	-7.7843E-1
50:	0.18838	1.1863E-7
75:	0.80992	7.7843E-1
90:	1.2978	1.4799E0
95:	1.5387	1.8998E0

GOODNESS OF FIT

CHI-SQUARE :	5.1937
DEG FREED:	3
SIGNIF :	0.15814
KOLM-SMIRN :	0.11667
SIGNIF :	0.28643
CRAMER-V M :	0.00021773
SIGNIF :	> .15
ANDER-DARL :	1.2178
SIGNIF :	> .15

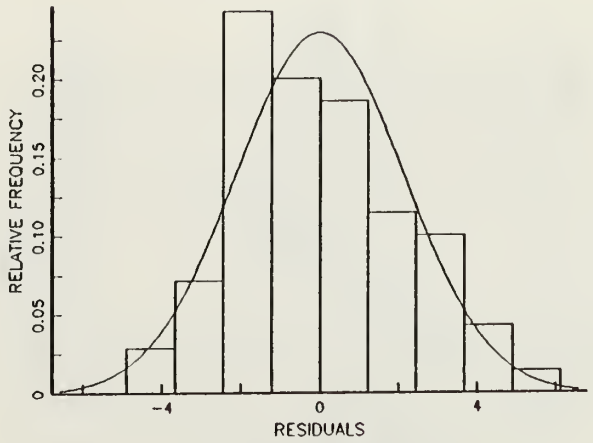
KS, AD, AND CV SIGNIF LEVELS NOT EXACT WITH ESTIMATED PARAMETERS

0.95 CONFIDENCE INTERVALS

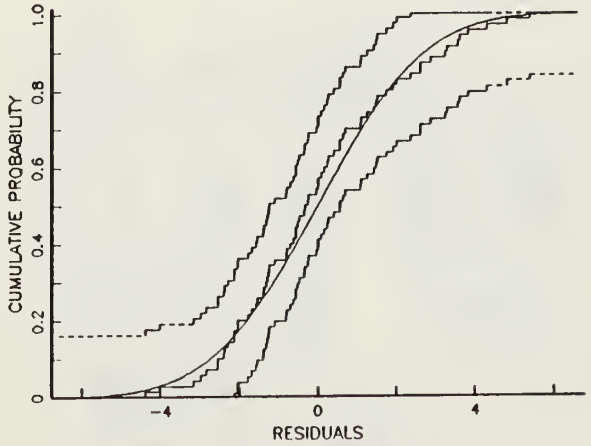
PARAMETER	ESTIMATE	LOWER	UPPER
MU	1.0854E-12	-0.27348	0.27348
SIGMA	1.1546E0	0.98997	1.3855

DATA SET D2A2 / B SENSOR / Y RESIDUALS

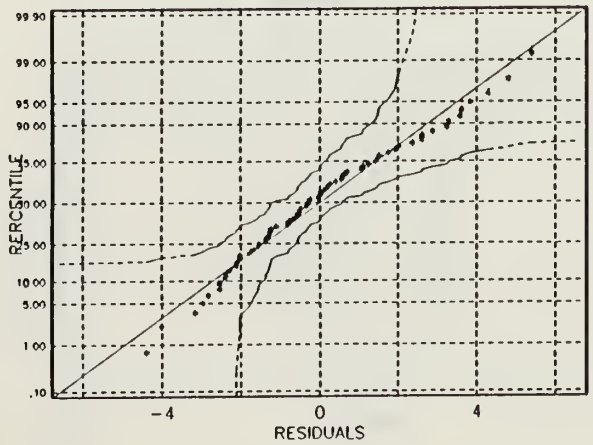
NORMAL DENSITY FUNCTION, N=70



NORMAL CUMULATIVE DISTRIBUTION FUNCTION, N=70



NORMAL PROBABILITY PLOT



NORMAL DISTRIBUTION

X : D2A2ZR[:5]
SELECTION : ALL
LABEL : RESIDUALS
SAMPLE SIZE: 70
MINIMUM : -4.399
MAXIMUM : 5.395
CENSORING : NONE
EST. METHOD: MAXIMUM LIKELIHOOD

	SAMPLE	FITTED
MEAN :	-3.8188E-14	-3.8188E-14
STD DEV :	2.1320E0	2.1320E0
SKEWNESS:	4.3995E-1	0.0000E0
KURTOSIS:	2.7143E0	3.0000E0

PERCENTILES	SAMPLE	FITTED
5:	-2.9759	-3.5077E0
10:	-2.5515	-2.7327E0
25:	-1.544	-1.4374E0
50:	-0.32757	2.1535E-7
75:	1.4302	1.4374E0
90:	3.2353	2.7327E0
95:	3.8311	3.5077E0

COVARIANCE MATRIX OF PARAMETER ESTIMATES

	MU	SIGMA
MU	0.06401	0
SIGMA	0	0.032469

GOODNESS OF FIT

CHI-SQUARE :	5.4013
DEG FREED:	3
SIGNIF :	0.14467
KOLM-SMIRN :	0.07736
SIGNIF :	0.79601
CRAMER-V M :	0.00010579
SIGNIF :	> .15
ANDER-DARL :	0.54149
SIGNIF :	> .15

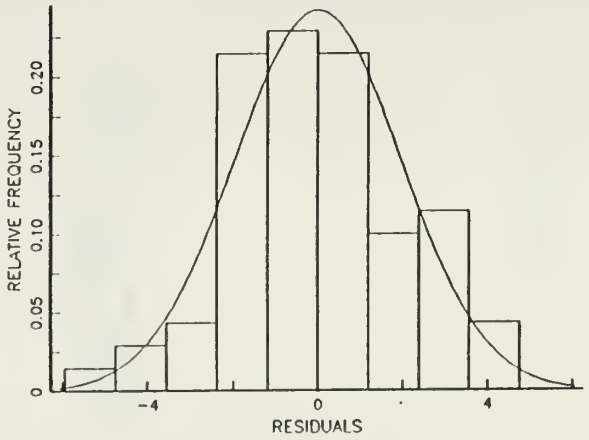
KS, AD, AND CV SIGNIF. LEVELS NOT EXACT WITH ESTIMATED PARAMETERS

0.95 CONFIDENCE INTERVALS

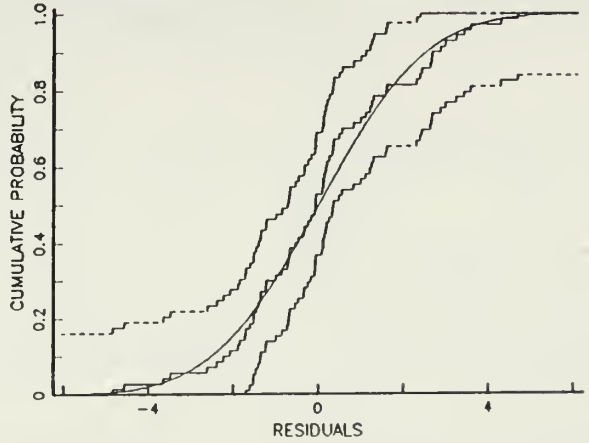
PARAMETER	ESTIMATE	LOWER	UPPER
MU	-3.8166E-14	-0.50495	0.50495
SIGMA	2.1320E0	1.828	2.5585

DATA SET D2A2 / B SENSOR / Z RESIDUALS

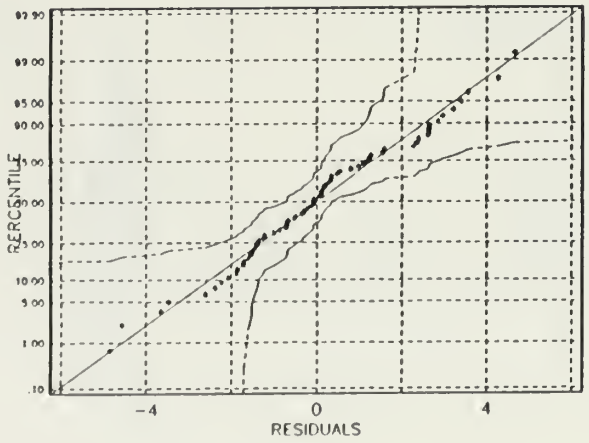
NORMAL DENSITY FUNCTION, N=70



NORMAL CUMULATIVE DISTRIBUTION FUNCTION, N=70



NORMAL PROBABILITY PLOT



NORMAL DISTRIBUTION

X : D2A2ZR[:0]
SELECTION : ALL
LABEL : RESIDUALS
SAMPLE SIZE: 70
MINIMUM : -4.834
MAXIMUM : 4.862
CENSORING : NONE
EST. METHOD: MAXIMUM LIKELIHOOD

	SAMPLE	FITTED
MEAN :	-0.3340E-14	-0.3340E-14
STD DEV :	1.9623E0	1.9623E0
SKEWNESS:	1.0522E-1	0.0000E0
KURTOSIS:	3.0088E0	3.0000E0

PERCENTILES	SAMPLE	FITTED
5:	-3.4776	-3.2284E0
10:	-2.2302	-2.5151E0
25:	-1.36-2	-1.3230E0
50:	-0.076473	1.9821E-7
75:	1.1948	1.3230E0
90:	2.8501	2.5151E0
95:	3.412	3.2284E0

COVARIANCE MATRIX OF PARAMETER ESTIMATES

	MU	SIGMA
MU	0.054223	0
SIGMA	0	0.027504

GOODNESS OF FIT

CHI-SQUARE :	4.587
DEG FREEO:	3
SIGNIF :	0.20487
KOLM-SMIRN :	0.10039
SIGNIF :	0.48071
CRAMER-V M :	0.00010369
SIGNIF :	> .15
ANDER-DARL :	0.51731
SIGNIF :	> .15

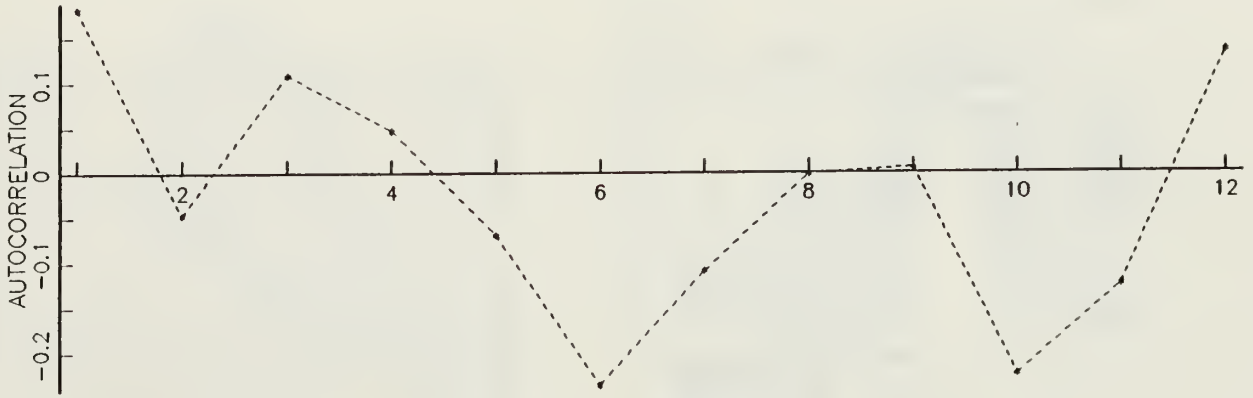
KS. AD. AND CV SIGNIF LEVELS NOT EXACT WITH ESTIMATED PARAMETERS

0.95 CONFIDENCE INTERVALS

PARAMETER	ESTIMATE	LOWER	UPPER
MU	-0.3340E-14	-0.48475	0.48475
SIGMA	1.9623E0	1.8825	2.3548

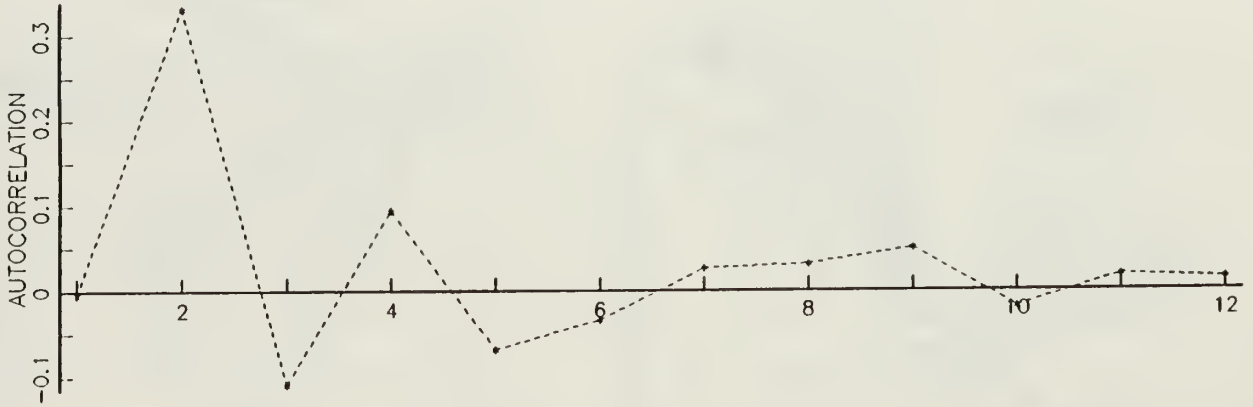
DATA SET D2A1 – TIME SERIES FOR LENGTH OF ERROR

SENSOR A



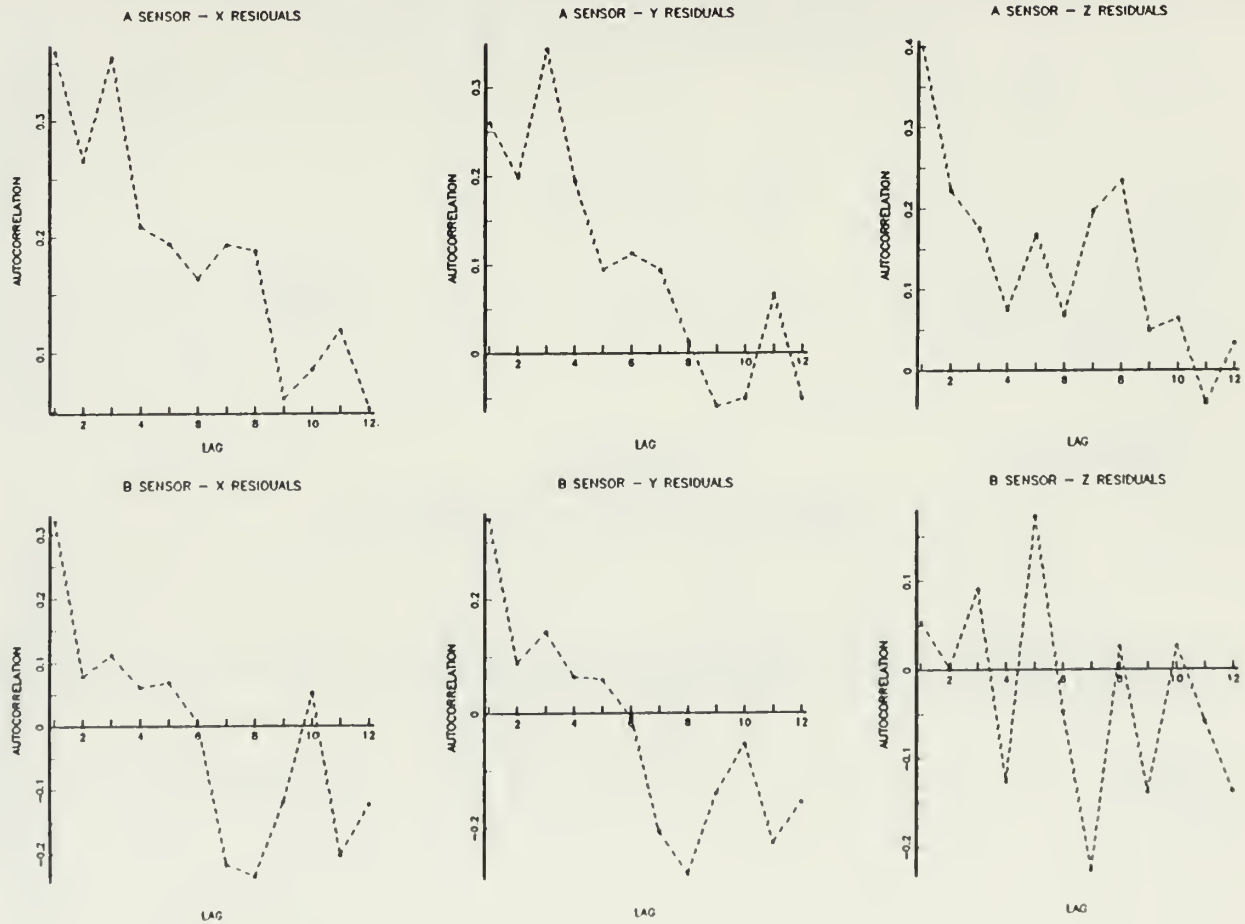
LAG : MAX LAG = .2 × NO. OF DATA POINTS

SENSOR B



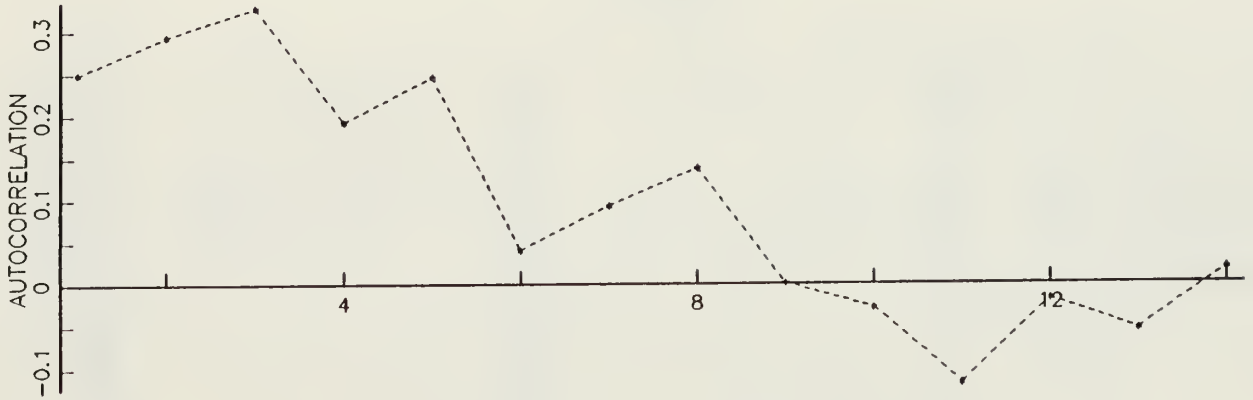
LAG : MAX LAG = .2 × NO. OF DATA POINTS

DATA SET D2A1 – TIME SERIES FOR RESIDUALS



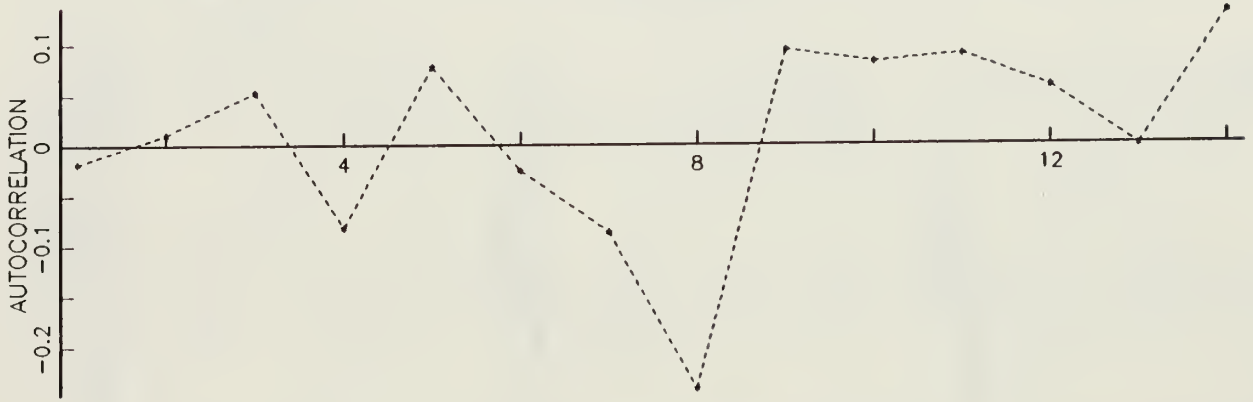
DATA SET D2A2 – TIME SERIES FOR LENGTH OF ERROR

SENSOR A



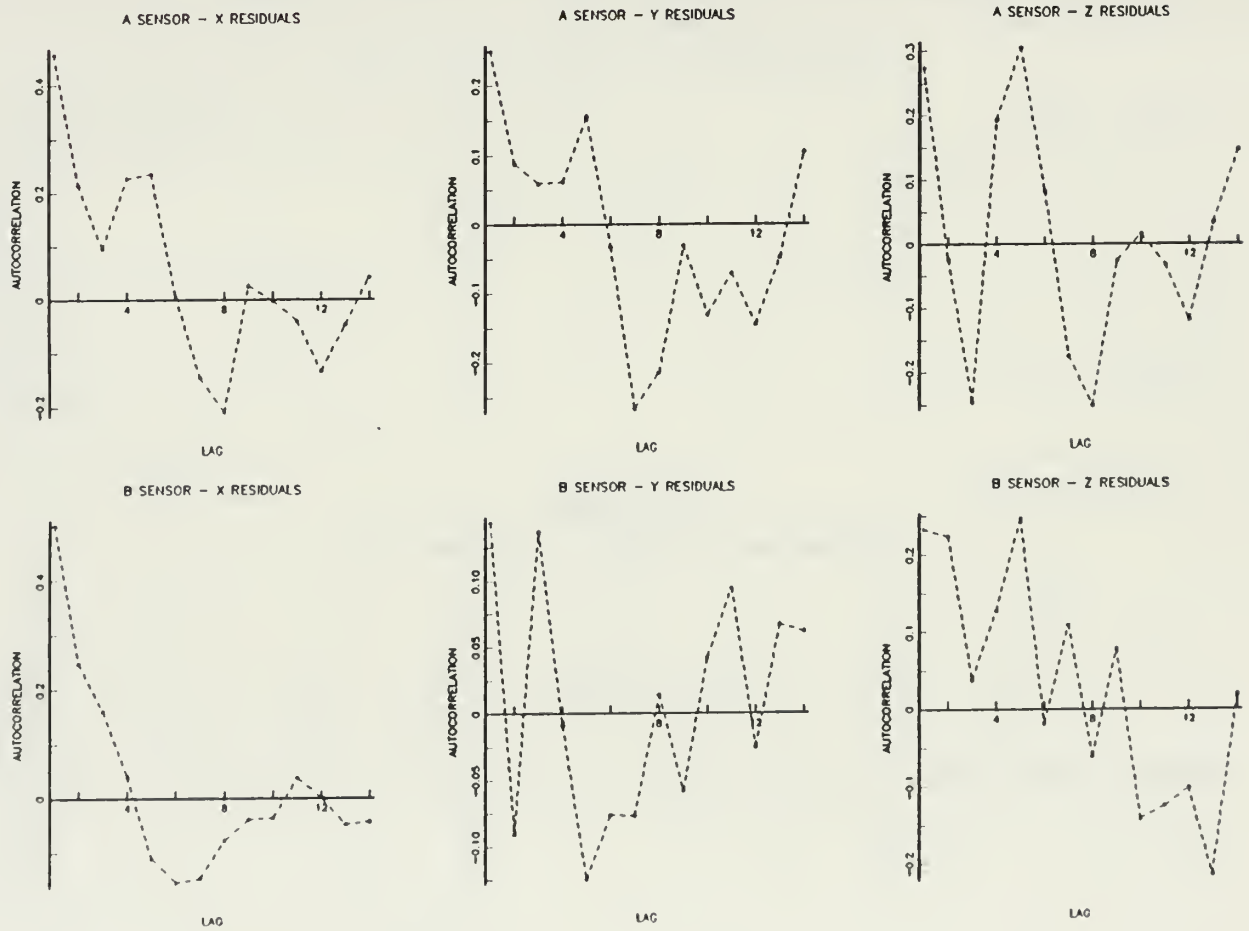
LAG : MAX LAG = .2 × NO. OF DATA POINTS

SENSOR B



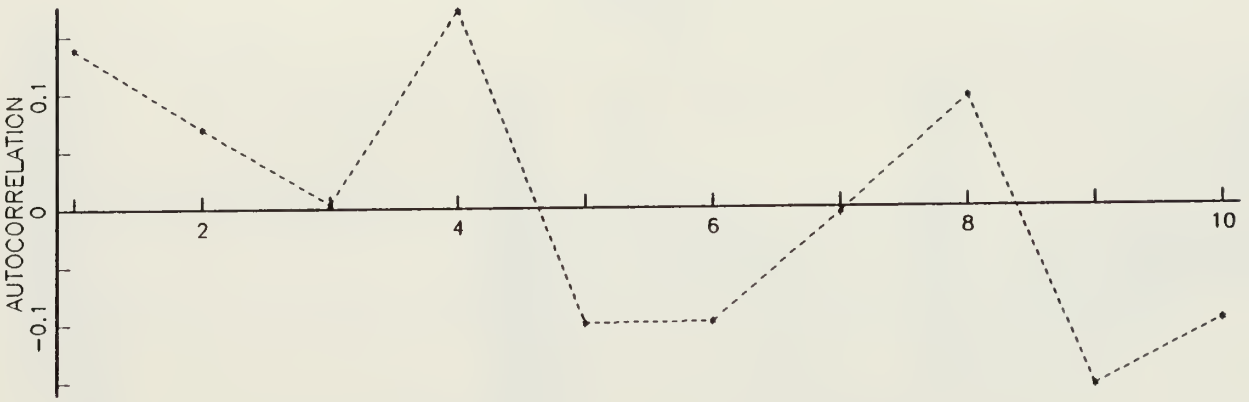
LAG : MAX LAG = .2 × NO. OF DATA POINTS

DATA SET D2A2 – TIME SERIES FOR RESIDUALS



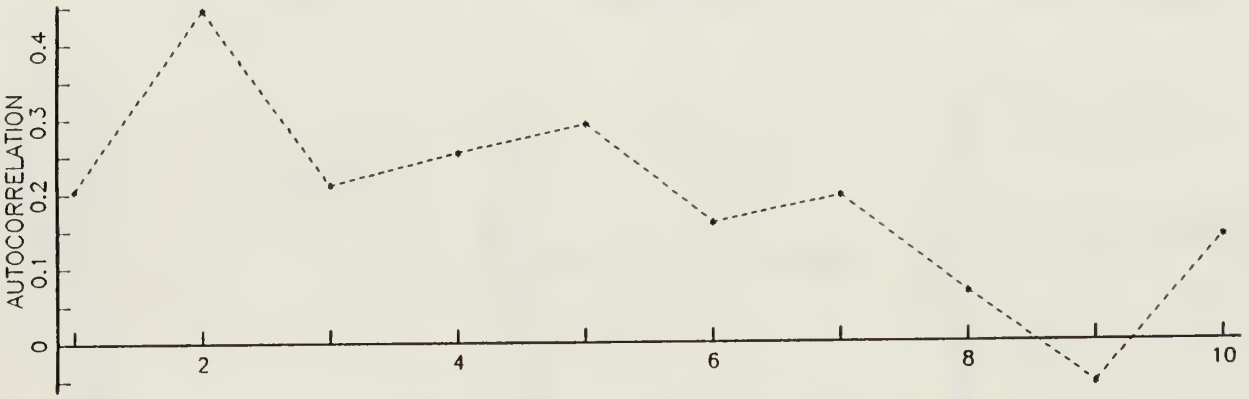
DATA SET D2B – TIME SERIES FOR LENGTH OF ERROR

SENSOR A



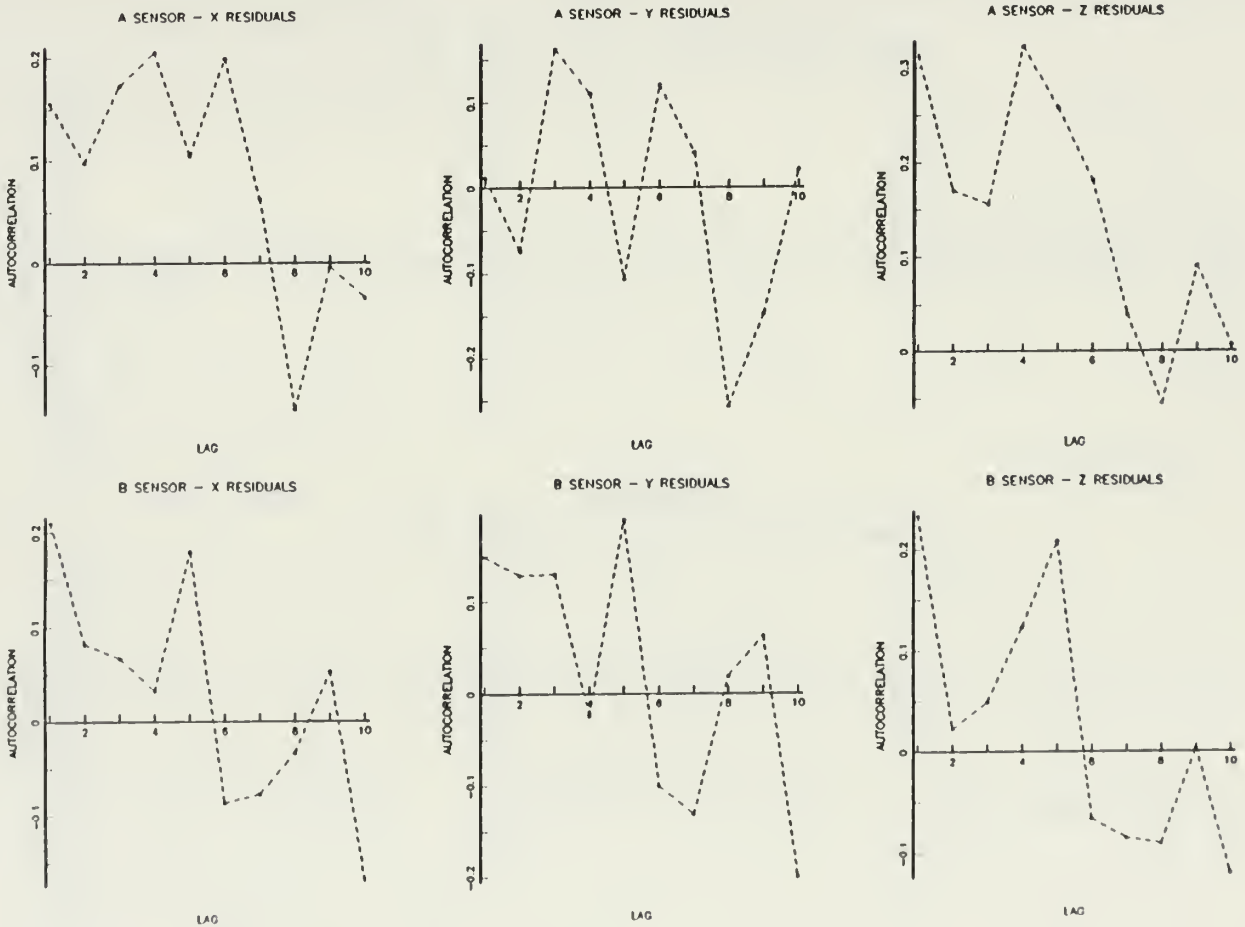
LAG : MAX LAG = .2 × NO. OF DATA POINTS

SENSOR B



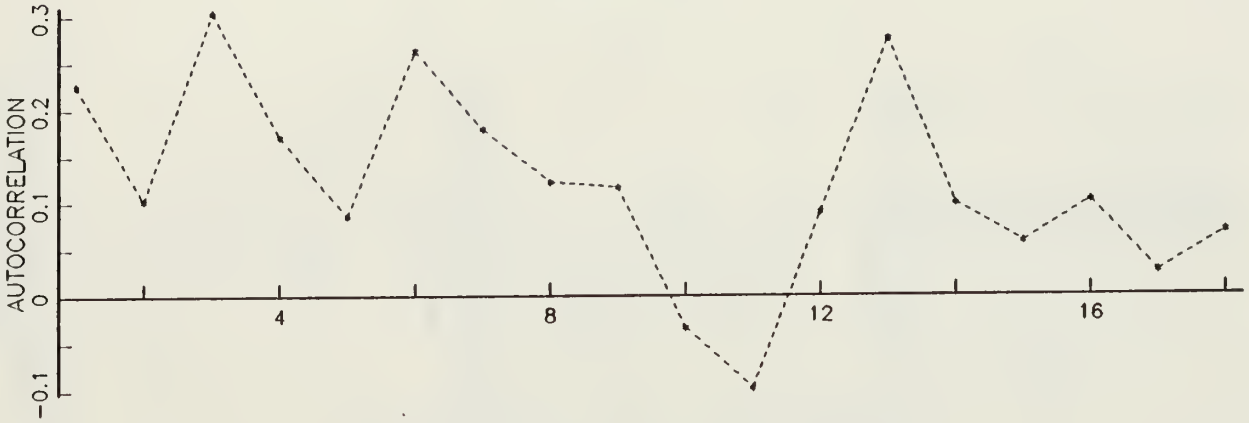
LAG : MAX LAG = .2 × NO. OF DATA POINTS

DATA SET D2B – TIME SERIES FOR RESIDUALS



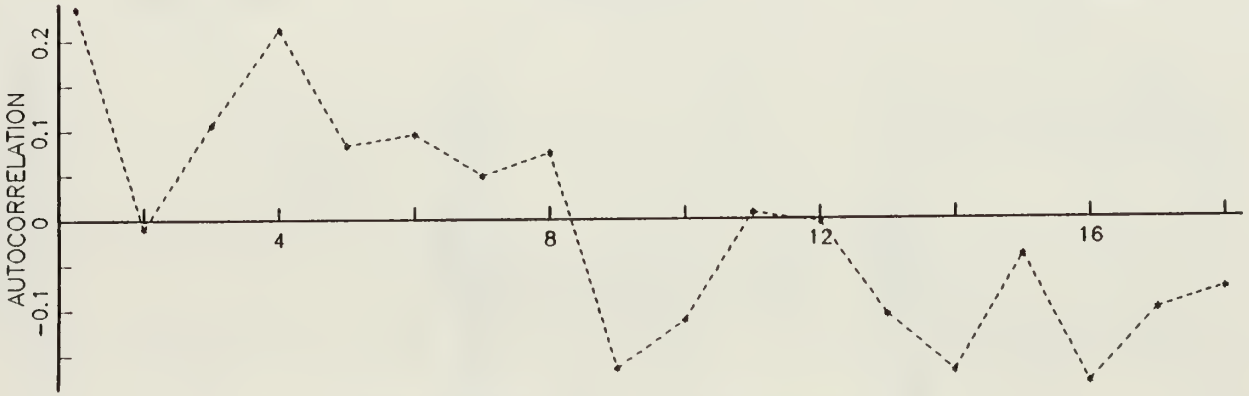
DATA SET D4 – TIME SERIES FOR LENGTH OF ERROR

SENSOR A



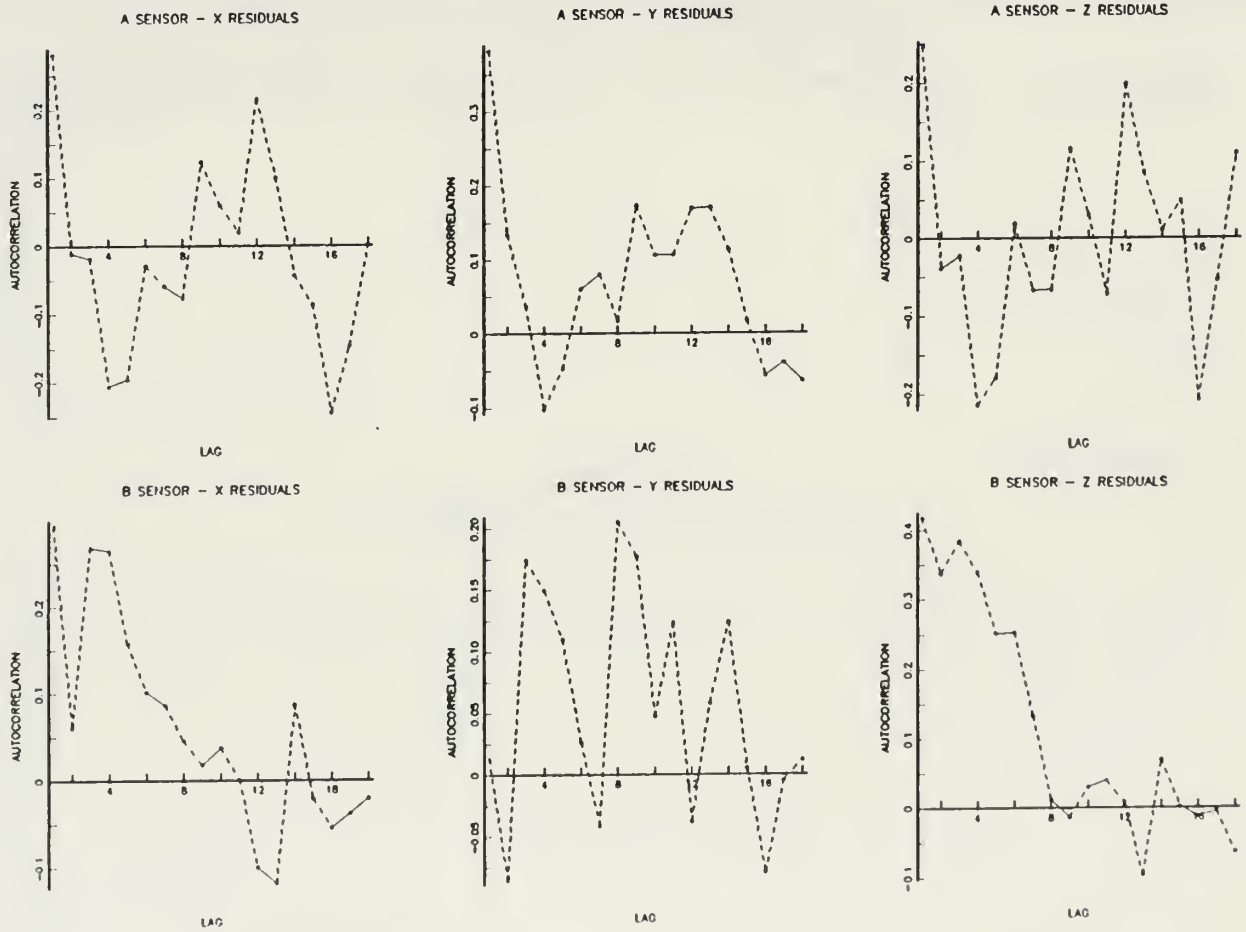
LAG : MAX LAG = .2 × NO. OF DATA POINTS

SENSOR B



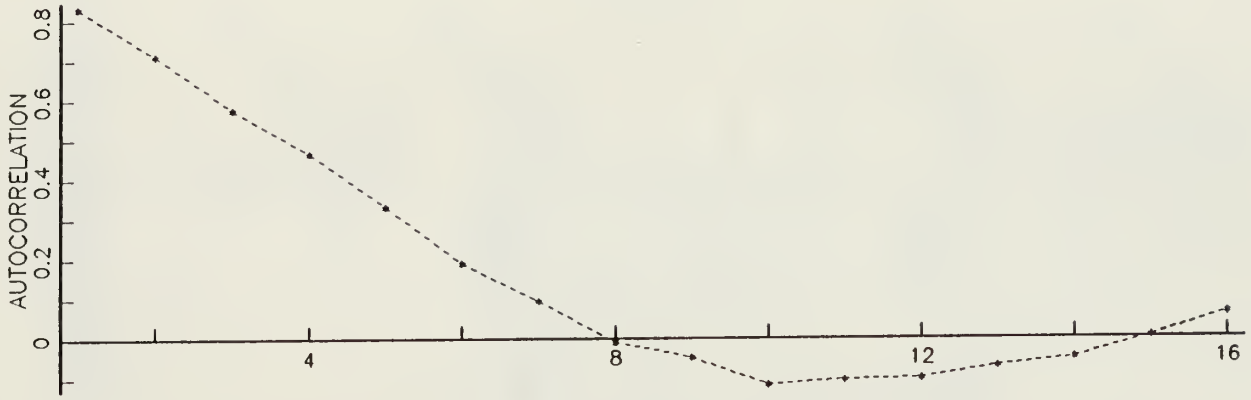
LAG : MAX LAG = .2 × NO. OF DATA POINTS

DATA SET D4 – TIME SERIES FOR RESIDUALS



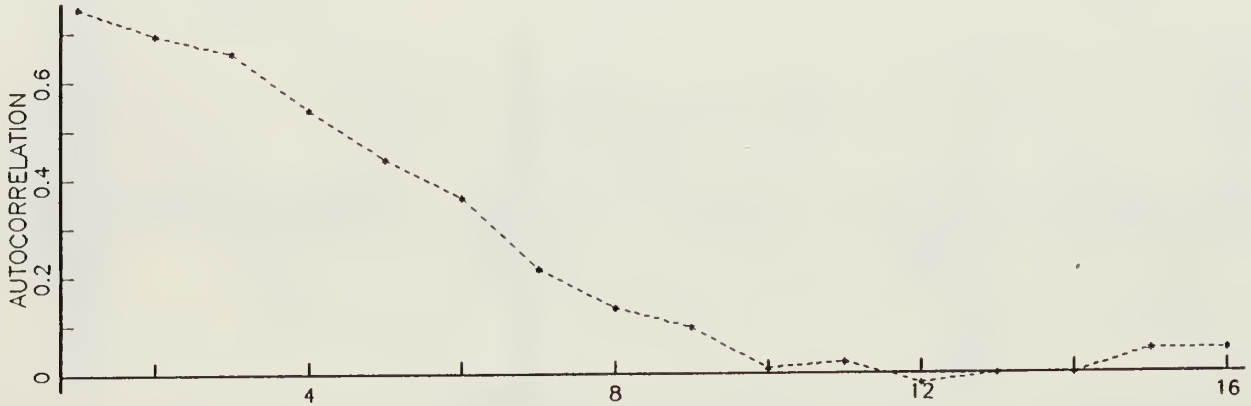
DATA SET D5A – TIME SERIES FOR LENGTH OF ERROR

SENSOR A



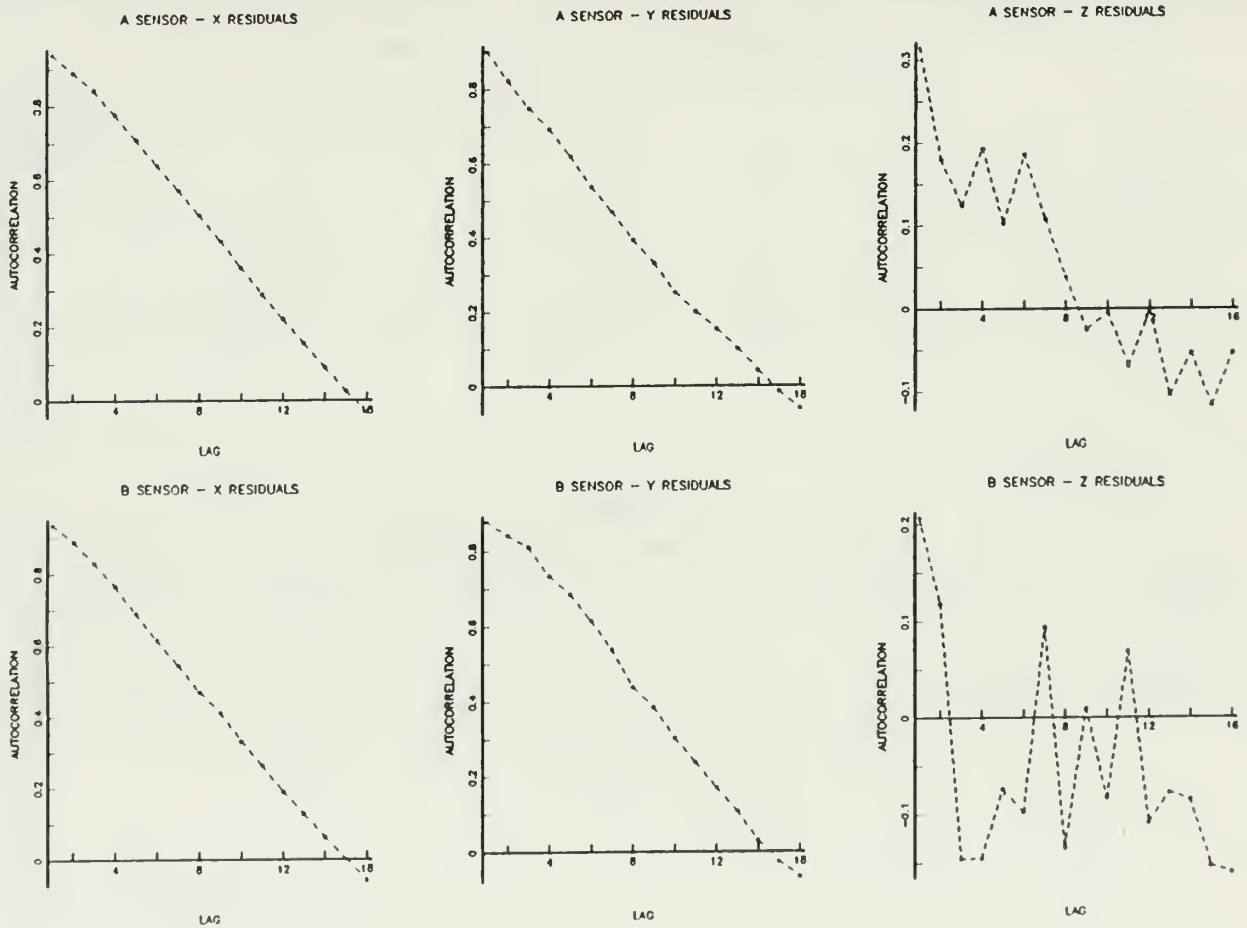
LAG : MAX LAG = .2 × NO. OF DATA POINTS

SENSOR B



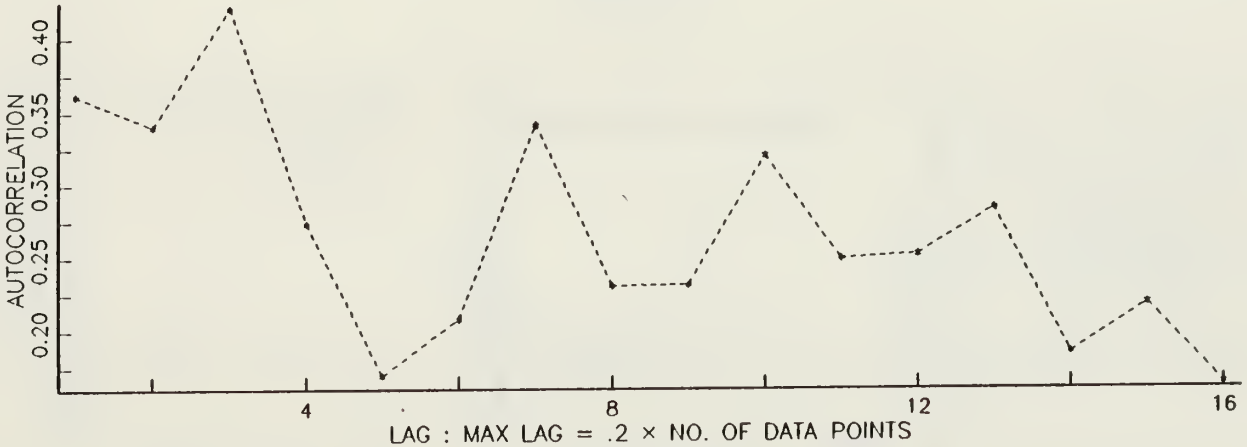
LAG : MAX LAG = .2 × NO. OF DATA POINTS

DATA SET D5A – TIME SERIES FOR RESIDUALS

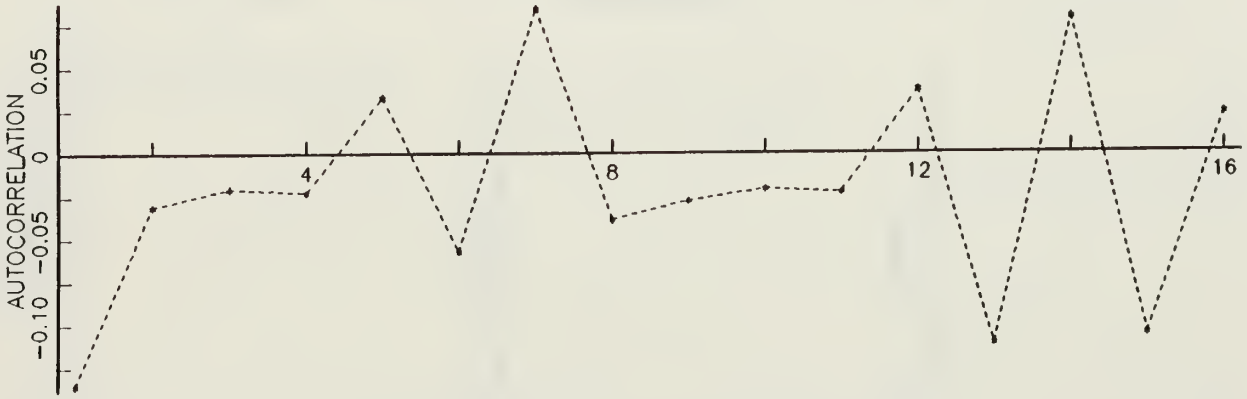


DATA SET D5B – TIME SERIES FOR LENGTH OF ERROR

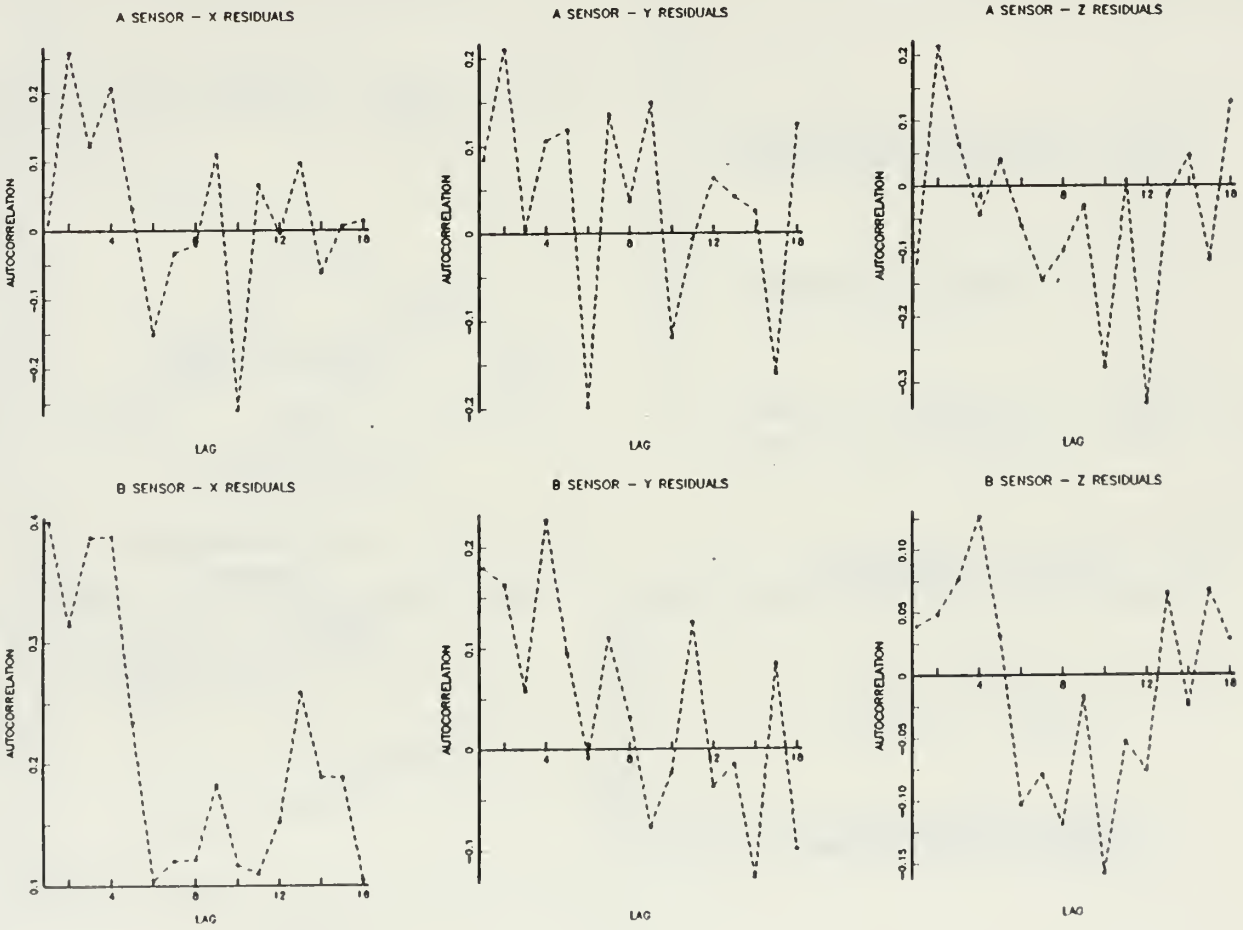
SENSOR A



SENSOR B

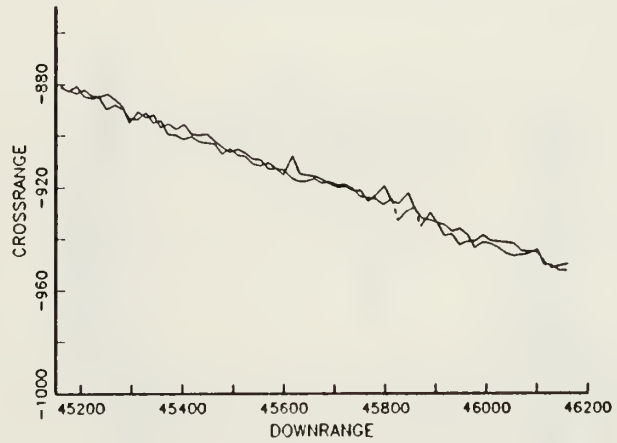


DATA SET D5B – TIME SERIES FOR RESIDUALS

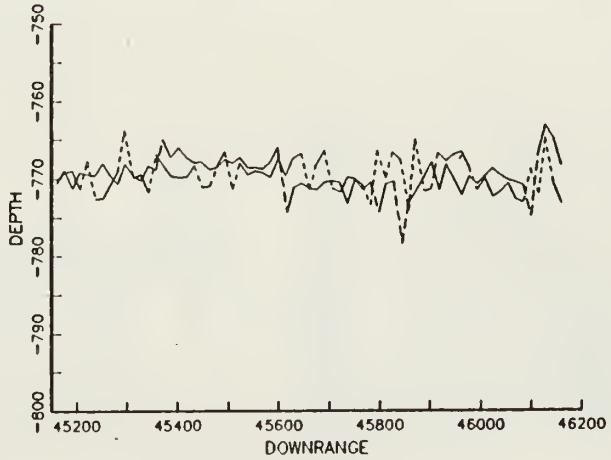


COMPARISON OF ORIGINAL AND SIMULATED TRACK

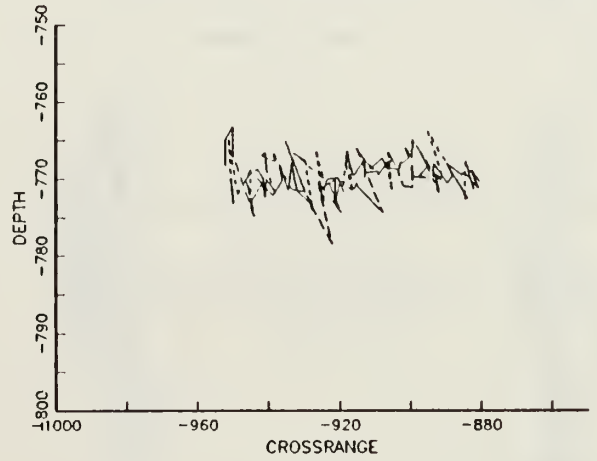
DOWNRANGE VS CROSSRANGE



DOWNRANGE VS DEPTH



CROSSRANGE VS DEPTH



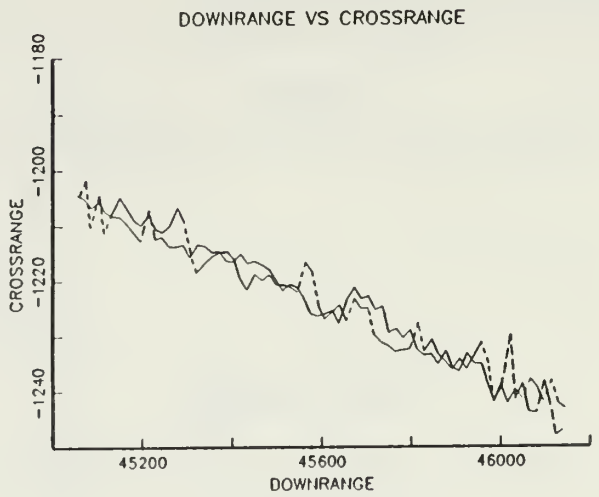
DATA SET D2A1

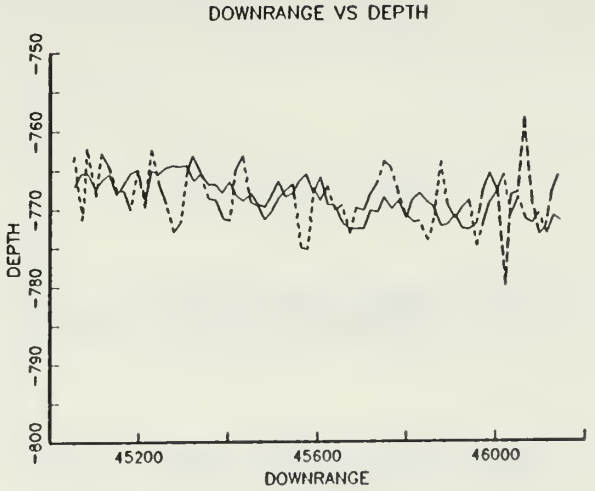
ORIGINAL TRACK

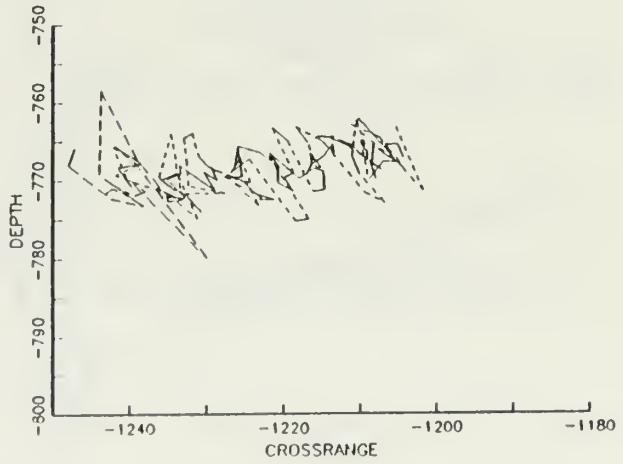
.....

SIMULATED TRACK

COMPARISON OF ORIGINAL AND SIMULATED TRACK







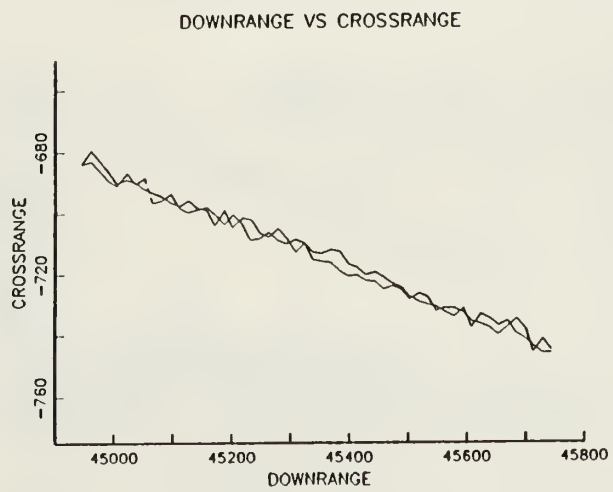
DATA SET D2A2

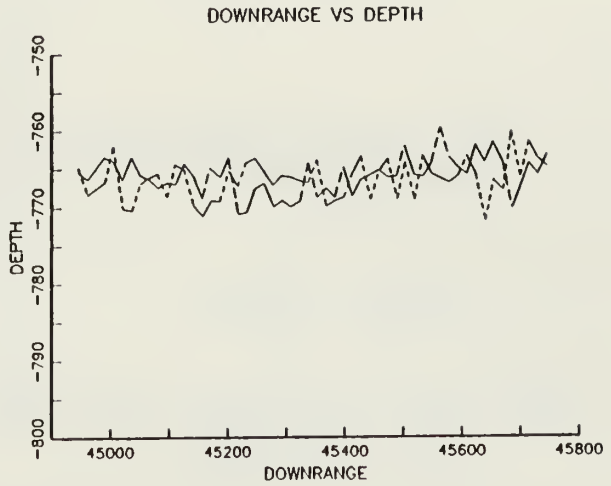
ORIGINAL TRACK

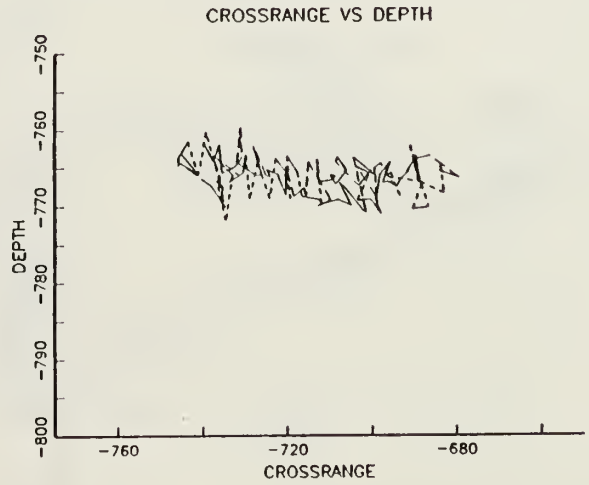
.....

SIMULATED TRACK

COMPARISON OF ORIGINAL AND SIMULATED TRACK





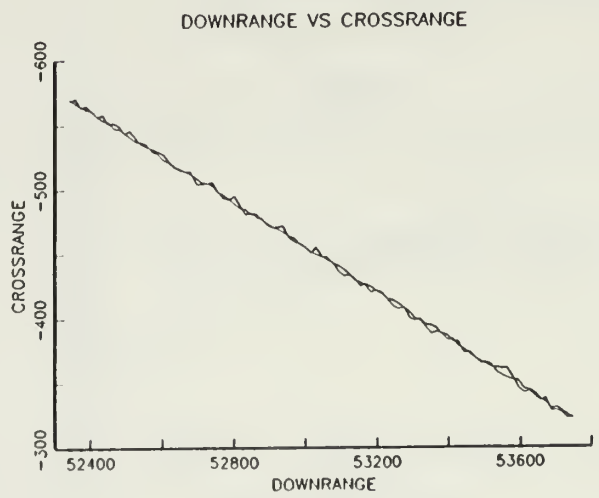


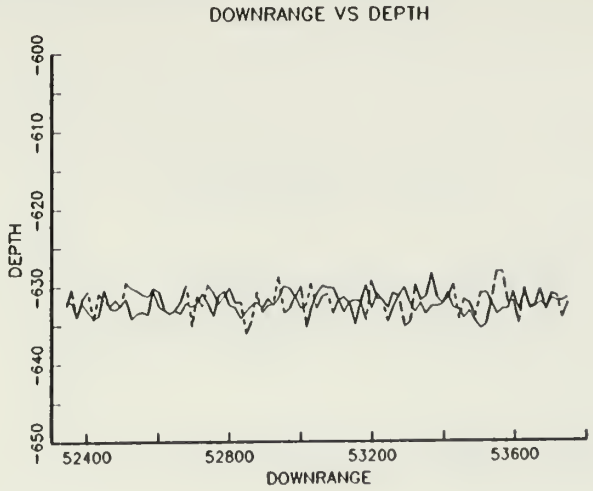
DATA SET D2B

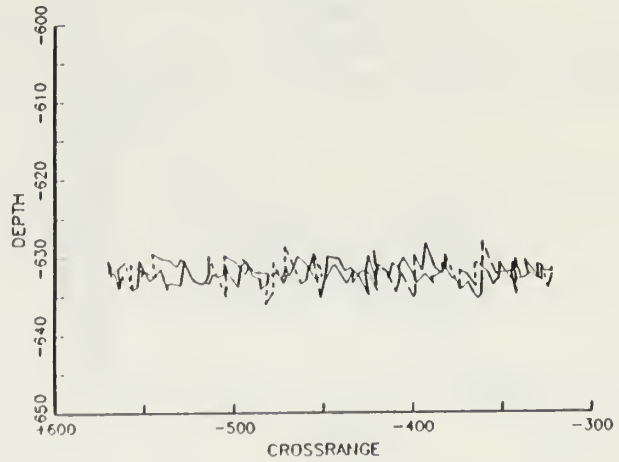
ORIGINAL TRACK

SIMULATED TRACK

COMPARISON OF ORIGINAL AND SIMULATED TRACK







DATA SET D4

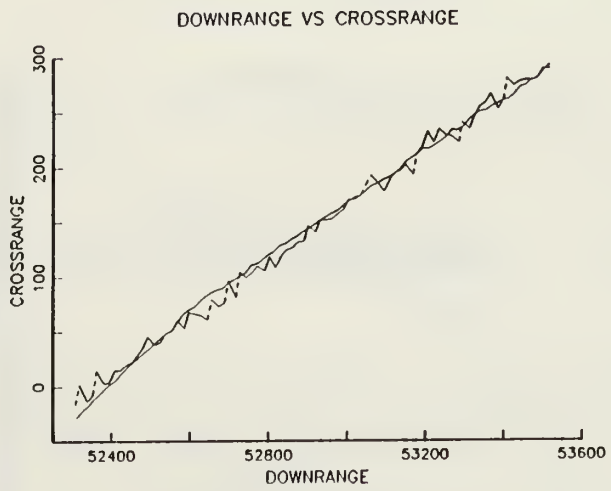
_ _ _ _ _ _ _ _ _ _ _

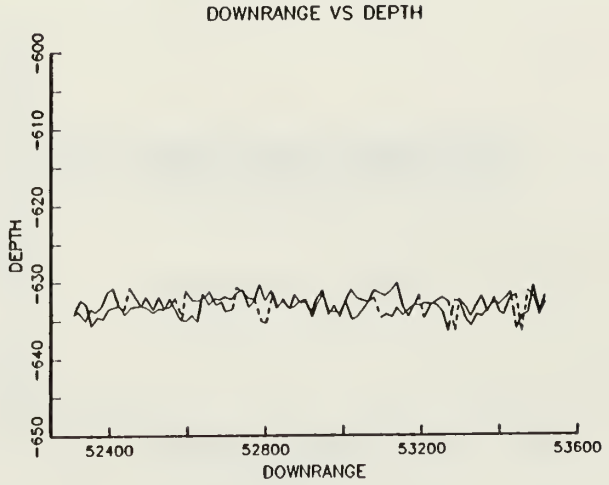
ORIGINAL TRACK

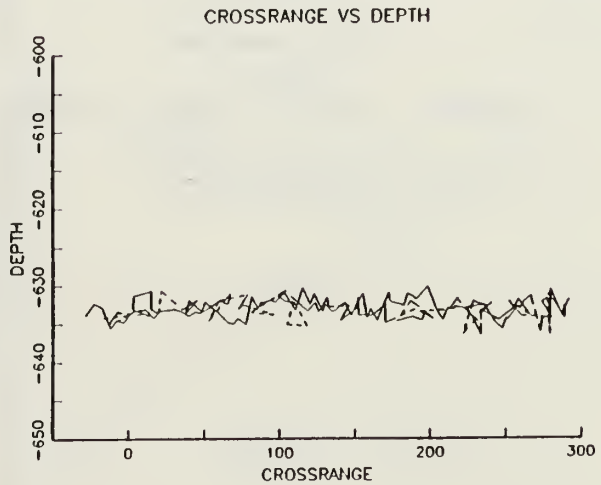
. . . . . .

SIMULATED TRACK

COMPARISON OF ORIGINAL AND SIMULATED TRACK







DATA SET D5A

_ _ _ _ _ _ _ _ _ _ _

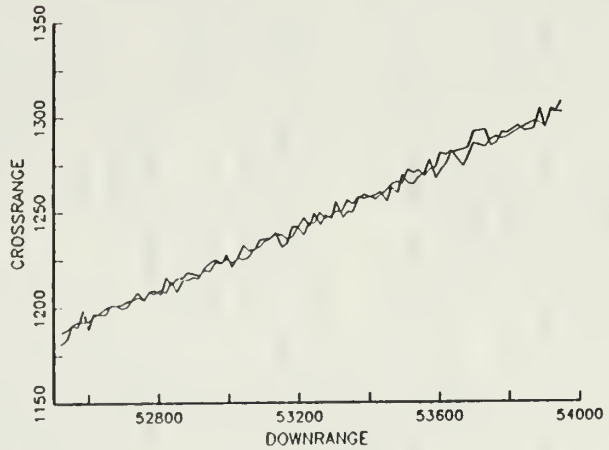
ORIGINAL TRACK

. . . . . .

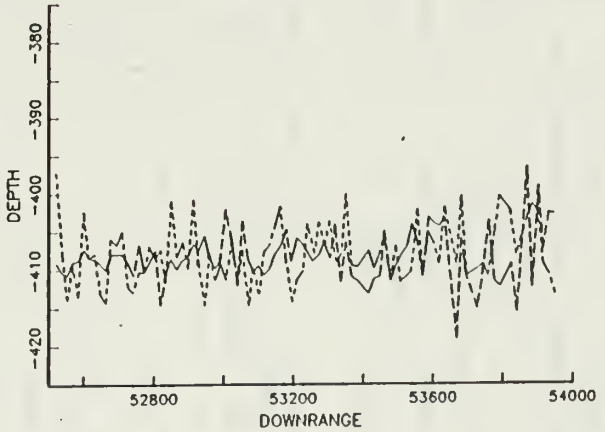
SIMULATED TRACK

COMPARISON OF ORIGINAL AND SIMULATED TRACK

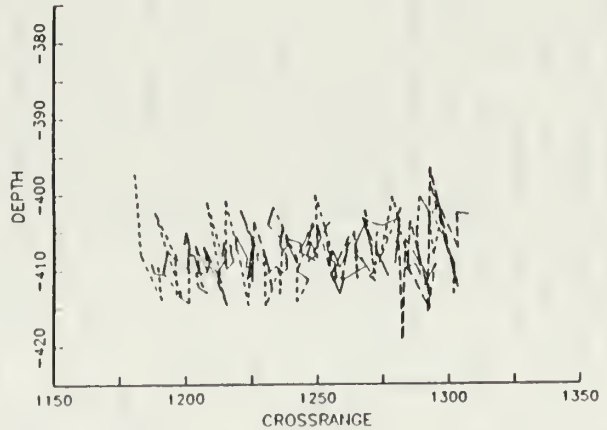
DOWNRANGE VS CROSSRANGE



DOWNRANGE VS DEPTH



CROSSRANGE VS DEPTH



DATA SET D5B

_ _ _ _ _ _ _ _ _ _

ORIGINAL TRACK

.....

SIMULATED TRACK

APPENDIX E

FORTRAN SUBROUTINES CONECT AND REDUCE

The program KEYMAIN requires that any array for which correction estimates are desired must be "connected" to the first input sensor array by one, or a series of, crossover data sets. For example, if an input crossover data set uses sensor arrays 5 and 6, while another uses arrays 6 and 10, all three of the arrays (5, 6 and 10) are connected. If, however, a first input data set uses arrays 7 and 9, a second input data set uses 12 and 10, while a third input data set uses 9 and 13, the arrays 7, 9 and 13 are connected, 12 and 10 are connected, but all five arrays do not form a single connected set. In this case, if 7 was the first array input to the program, correction estimates could not be made for arrays 10 and 12. The subroutine CONECT checks to see that connectedness exists in the input data before KEYMAIN is allowed to continue.

KEYMAIN allows 3 options if CONECT discovers that the arrays of the input data sets are not connected. One option is to quit, in which case the program terminates. Another option is to add more data so that all the arrays are connected. In the second example above, for instance, if a crossover data set using arrays 9 and 12 was input into the program, all 5 arrays would then be connected and KEYMAIN

would continue. A third option is to continue with KEYMAIN, but use only the first connected set, that is, all the arrays that are connected to the left array of the first input crossover data set. This option presents special problems because it requires a reduction of the data structures that have been built as each data set was input. The subroutine REDUCE does this data structure reduction and is called from KEYMAIN only when this third option is selected.

KEYMAIN passes to CONECT two pieces of information. The first is the variable R1 that represents the number of data sets that were input into KEYMAIN. The second piece of information is a 2 x R1 matrix, IND2, that contains the number of the left and right sensors of the R1 crossover data sets. Row 1 contains the left sensor numbers, row 2 the right. CONECT performs its connectedness check by starting a variable length list that contains the array numbers of those arrays that are connected. The list starts with only two entries, the left and right sensor arrays of the first crossover data set. These are connected, and the left sensor is the "root" to which all arrays should connect. Elements are added to the list by sequencing through the list from the beginning, and adding to the list any array numbers for IND2 that (1) are not yet on the list and (2) are connected by a crossover data set to an array that is already on the list. If, after sequencing through

the variable length list, there are arrays remaining in IND2 that never were put on the list, those arrays were not a part of the first connected set. If all arrays in IND2 were included on the list, all the arrays were connected. After the first list is exhausted, CONECT repeats this procedure as many times as there are disjoint sets of arrays, starting each new list with the arrays of a data set not in any previous set. CONECT informs the user of the individual sets of connected sets of arrays and raises a flag to alert the user if all sets were not connected.

If all the arrays in the input crossover data sets were not connected, the user has three options to proceed, described earlier. If he chooses to continue using the first set of connected arrays, REDUCE is called to pare the data strauctures built up during the data input process. In particular, the R1 x 3 x 3 array CROSSA and the R1 x 6 matrix mean need to be reduced to contain only those elements that correspond to the data sets in the first connected set. CROSSA, which has R1 "pages" of 3 x 3 matrices of crossproduct deviations from the mean, needs to have those pages removed that correspond to every crossover data set in the original input not connected to the first array. MEAN, which has R1 rows of the colunm averages for each data set, needs to have those rows removed that correspond to data sets not connected to the first array.

The variable R1 itself will be reduced to reflect this smaller subset of connected array pairs.

CONECT stores the information that indicates which data sets in IND2 are connected to the first array. This information is passed to REDUCE through KEYMAIN. REDUCE reforms the data structures to reflect the smaller number of data sets now being considered by KEYMAIN.

CONECT also provides the matrix IND1, a K x R1 matrix that is used elsewhere in KEYMAIN. The R1 columns represent the crossover data sets input into KEYMAIN. The variable K represents the number of individual arrays in the R1 data sets, each row representing a separate array. For each column in IND1, the entries are all 0 except in the rows that represent the left and right sensor for that column's data set. If the row corresponds to the left array, the value is 1. If it corresponds to the right array, the value is 2. If REDUCE is called, some columns (representing crossover data sets) and rows (representing individual sensors) of IND1 need to be omitted. REDUCE reforms IND1 to its smaller size.

APPENDIX F

FORTRAN LISTING FOR SUBROUTINE CONECT

```
      SUBROUTINE CONECT (OUT,R1,IND2,K,IND1,IA,TESTC,IND2R,
     *                   DATSET)
C
C*************************************************************
C   This subroutine checks for the connectedness of the
C   input data sets. If the problem is connected then the
C   user is informed and the array pairs are printed on the
C   screen; if not connected, then the user is prompted to
C   select one of three options - quit, add conecting data
C   sets, or run the program using the first connected set
C   that was input.              Gygax - July 1985
C*************************************************************
C
C      ...Variable declarations.
C
      INTEGER*4 R1,K,IND2(2,30),IND1(30,30),I,J,IA(30),FIRST
      INTEGER*4 LIST(30),BEGIN,HALT,DISCON,L,M,O,TESTC,OUT,
      INTEGER*4 DATSET(30),COUNT,SAVE(2,30), IND2R(2,30)
C
C      ...Initialize the values of FIRST and COUNT:
C
      FIRST = 0
      COUNT = 0
C
C      ...Make vector IA = list of all arrays (w/o repeats)
C         in IND2 and get the value for K = # of individual
C         arrays.
C
      IA(1) = IND2(1,1)
      IA(2) = IND2(2,1)
      K = 3
      IF (R1 .EQ. 1) GOTO 60
      DO 50 I = 1,R1
         DO 40 J = 1,2
            M = K - 1
            DO 30 L = 1,M
               IF (IND2(J,I) .EQ. IA(L)) GOTO 40
 30         CONTINUE
            IA(K) = IND2(J,I)
            K = K + 1
 40      CONTINUE
 50      CONTINUE
 60   K = K - 1
C
```

```
      WRITE(OUT,*)'R1   ',R1
      WRITE(OUT,*)'K    ',K
C     ...For each column of IND1 (columns correspond  to
C        data sets) the entries are all zero except for
C        the row that corresponds to the left array (= 1)
C        and the right array (= 2).
C
      DO 80 I = 1,R1
         DO 70 J = 1,K
            IND1(J,I) = 0
            IF (IND2(1,I) .EQ. IA(J)) IND1(J,I) = 1
            IF (IND2(2,I) .EQ. IA(J)) IND1(J,I) = 2
 70      CONTINUE
 80   CONTINUE
C
C     ...Check to see if all the arrays are connected.
C
      TESTC = 1
      LIST(1) = -IA(1)
      DO 131 I = 1,R1
         IF (IND2(1,I) .EQ. -LIST(1)) IND2(1,I) = -IND2(1,I)
         IF (IND2(2,I) .EQ. -LIST(1)) IND2(2,I) = -IND2(2,I)
 131  CONTINUE
      BEGIN = 1
      HALT = 1
 140  IF (.NOT. (BEGIN .LE. HALT)) GOTO 170
      NODE = LIST(BEGIN)
      BEGIN = BEGIN + 1
      DO 150 I = 1,R1
       IF (.NOT.((NODE.EQ.IND2(1,I)).AND.(IND2(2,I).GT.0)))
     *           GOTO 150
        HALT = HALT + 1
        LIST(HALT) = -IND2(2,I)
        DO 141 J = 1,R1
         IF (IND2(1,J).EQ.-LIST(HALT)) IND2(1,J)=-IND2(1,J)
         IF (IND2(2,J).EQ.-LIST(HALT)) IND2(2,J)=-IND2(2,J)
 141    CONTINUE
 150  CONTINUE
      DO 160 I = 1,R1
       IF (.NOT.((NODE.EQ.IND2(2,I)).AND.(IND2(1,I).GT.0)))
     *           GOTO 160
        HALT = HALT + 1
        LIST(HALT) = -IND2(1,I)
        DO 151 J = 1,R1
         IF (IND2(1,J).EQ.-LIST(HALT)) IND2(1,J)=-IND2(1,J)
         IF (IND2(2,J).EQ.-LIST(HALT)) IND2(2,J)=-IND2(2,J)
 151    CONTINUE
 160  CONTINUE
      GOTO 140
 170  CONTINUE
C
C     ...Print out the matched pairs.
```

```
C
      DISCON = 0
      WRITE(OUT,230)
      DO 200 I = 1,R1
      IF (IND2(1,I) .LT. 0) GOTO 190
      IF (IND2(1,I) .EQ. 0) GOTO 200
      IF ((IND2(1,I) .GT. 0) .AND. (DISCON .EQ. 1)) GOTO 200
      FIRST = FIRST + 1
      DISCON = 1
      TESTC = 0
      BEGIN = 1
      HALT = 1
      LIST(1) = IND2(1,I)
      GOTO 200
 190  WRITE(OUT,240) -IND2(1,I),-IND2(2,I)
      IF ((FIRST.EQ.0).OR.((FIRST.EQ.1).AND.(DISCON.EQ.1)))
     *                    THEN
         COUNT = COUNT + 1
         IND2R(1,COUNT) = -IND2(1,I)
         IND2R(2,COUNT) = -IND2(2,I)
         DATSET(COUNT) = I
      END IF
      SAVE(1,I) = -IND2(1,I)
      SAVE(2,I) = -IND2(2,I)
      IND2(1,I) = 0
      IND2(2,I) = 0
 200  CONTINUE
      IF (DISCON .EQ. 1) GOTO 140
      DO 220 I = 1,R1
         IND2(1,I) = SAVE(1,I)
         IND2(2,I) = SAVE(2,I)
 220  CONTINUE
      RETURN
 230  FORMAT(1X,'THE FOLLOWING PAIRS ARE CONNECTED :')
 240  FORMAT(1X,14I5)
      END
```

APPENDIX G

FORTRAN LISTING FOR SUBROUTINE REDUCE

```
      SUBROUTINE REDUCE (CROSSA,MEAN,R1,K,IND1,IA,
     *                                   IND2R,DATSET)
C
C*****************************************************************
C     This is a specialized subroutine that is used  when
C     option three is invoked as a result of a failed con-
C     nectedness test. The disconnected data sets must be
C     removed from the variables CROSSA and MEAN, and other
C     program supporting variables must be adjusted.
C     Gygax - July 1985
C*****************************************************************
C
C       ... Variable declarations.
C
      INTEGER*4 R1,K,IND1(30,30),IA(30),IND2R(2,30),I,J,L,
     2          M,DATSET(30)
C
      REAL*8 CROSSA(30,3,3),MEAN(30,6)
C
C       ... Compute the new, reduced R1:
C
      DO 10 I = 1,30
         IF (IND2R(1,I) .EQ. 0) GOTO 20
 10   CONTINUE
 20   R1 = I - 1
C
C       ... Make new, reduced vector IA = list of all arrays
C           in IND2R w/o repeats. Also, compute a new K.
C
      IA(1) = IND2R(1,1)
      IA(2) = IND2R(2,1)
      K = 3
      IF (R1 .EQ. 1) GOTO 60
      DO 50 I = 1,R1
         DO 40 J = 1,2
            M = K - 1
            DO 30 L = 1,M
               IF (IND2R(J,I) .EQ. IA(L)) GOTO 40
 30         CONTINUE
            IA(K) = IND2R(J,I)
            K = K + 1
 40      CONTINUE
 50   CONTINUE
 60   K = K - 1
```

```
C
C      ... Remake the reduced matrix IND1 - for each column
C          in  IND1  (corressponding  to a  data  set)  the
C          entries are zero except for the entries corres-
C          ponding to the left array (= 1) and the right
C          array (= 2).
C
      DO 80 I = 1,R1
         DO 70 J = 1,K
            IND1(J,I) = 0
            IF (IND2R(1,I) .EQ. IA(J)) IND1(J,I) = 1
            IF (IND2R(2,I) .EQ. IA(J)) IND1(J,I) = 2
 70      CONTINUE
 80   CONTINUE
C
C      ... Reduce the arrays CROSSA and MEAN to account
C          for the removed data sets.
C
      DO 120 I = 1,R1
         DO 90 J = 1,6
            MEAN(I,J) = MEAN(DATSET(I),J)
 90      CONTINUE
         DO 110 J = 1,3
            DO 100 L = 1,3
               CROSSA(I,J,L) = CROSSA(DATSET(I),J,L)
 100        CONTINUE
 110     CONTINUE
 120  CONTINUE
      RETURN
      END
```

LIST OF REFERENCES

1. Main, C. D., Alternative Models for Calculation of Elevation Angles and Ray Transit Times for Ray Tracing of Hydrophonic Tracking Data, p.76, Master of Science Thesis, Naval Postgraduate School, Monterey, California, September 1984.

2. Larson H. J.,Introduction to Probability Theory and Statistical Inference, John Wiley & Sons, Inc., 1982.

3. Brownlee, K. A., Statistical Theory and Methodology in Science and Engineering, John Wiley & Sons, Inc., 1960.

4. IMSL, Problem Solving Software Systems, 1985.

BIBLIOGRAPHY

Naval Postgraduate School Report NPS55-83-031PR, Task 83-7: Analysis of Tracking Data, by R. R. Read, October 1983.

Read, R. R., New Algorithm to Estimate Displacement and Orientation Corrections, letter report to Naval Undersea Weapons Engineering Station (NUWES), October 1984.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22304-6145	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93943-5100	2
3.	Commanding Officer Attn : Mr. G. Krancus, Code 50 Naval Undersea Weapons Engineering Station (NUWES) Keyport, Washington 98345	3
4.	Professor R. R. Read, Code 55Re Operations Research Department Naval Postgraduate School Monterey, California 93943-5100	2
5.	Professor J. D. Esary, Code 55Ey Operations Research Department Naval Postgraduate School Monterey, California 93943-5100	2
6.	Mr. Robert W. Corpening 1902 Carolina Ave. Lynn Haven, Florida 32444	1
7.	Capt. Rex Gygax, USN (Ret) P.O. Box 776 Cloudcroft, New Mexico 88317	1
8.	LCdr. Gene Gygax 1 Biddle Lane Monterey, California 93940-6201	1

Thesis
G99 Gygax
c.1 The simulation of remotely measured paths of underwater vehicles for the purpose of monitoring the calibration of test ranges.